UNIPÊ - CENTRO UNIVERSITÁRIO DE JOÃO PESSOA
CURSO: CIÊNCIA DA COMPUTAÇÃO
TURMA: P1-B

Relatório de Projeto da Disciplina Técnicas e Desenvolvimento de Algoritmo

Jogo da Velha em C

Alunas:
Anna Maria Albino de Oliveira (RGM 35341581**)**
Maria Eduarda Zumbé Gomes (RGM 34964321)
Milena Azevedo Crispim (RGM 35059281)

João Pessoa - PB
2023

Introdução

O jogo da velha, conhecido internacionalmente como "Tic-Tac-Toe", é um jogo de estratégia simples, mas divertido, ideal para duas pessoas. O objetivo do jogo é ser o primeiro jogador a conseguir alinhar três de seus próprios símbolos (geralmente um 'X' ou um 'O') verticalmente, horizontalmente ou diagonalmente em uma grade de 3x3.

O jogo é jogado em um tabuleiro quadrado, dividido em 9 espaços menores. Não é necessário equipamento especial; pode ser jogado em um papel, quadro-negro, ou até digitalmente. No caso do presente jogo apresentado neste relatório, de forma digital, foi elaborado em linguagem C.

Regras do Jogo:

- Início do Jogo: Dois jogadores escolhem seus símbolos. Tradicionalmente, um jogador usa 'X' e o outro usa 'O'.

- Movimentos: Os jogadores se revezam para colocar seus símbolos em um espaço vazio na grade. O jogador com 'X' geralmente joga primeiro.

- Ganhando o Jogo: Vence o jogador que conseguir alinhar três de seus símbolos em linha reta, horizontal, vertical ou diagonal primeiro. Se todos os nove espaços são preenchidos sem que um jogador consiga alinhar três símbolos, o jogo termina empatado.

Resultados

A programação do jogo foi feita na linguagem C, por meio do Dev C + +.

Em primeiro lugar, pensamos nos requisitos principais do jogo, para assim começarmos o protótipo do código.

Estrutura de Dados

1. Estrutura Jogador:

- nome[50]: Armazena o nome do jogador. É um array de caracteres com capacidade para 49 caracteres mais o caractere nulo de terminação.
- vitórias: Armazena o número de vitórias do jogador. É um inteiro.

```
struct Jogador{
    char nome[50];
    int vitorias;
};
```

Funções

1. Função' lerNumero':

Esta função lê um número inteiro da entrada padrão (stdin). Se a entrada não for um número inteiro, ela exibe uma mensagem de erro e limpa o buffer de entrada (usando getchar() em um loop até encontrar um caractere de nova linha). Esta função é útil para validar entradas do usuário.

```
int lerNumero() { // Função para tratar erros do usuário
    setlocale (LC_ALL, "portuguese");
    int numero;
    while (scanf("%d", &numero) != 1) {
        printf("\n Opção Inválida! Digite um número: \n ");
        // Limpa o buffer de entrada
        while (getchar() != '\n');
    }
    return numero;
}
```

Função Principal main

1. Declaração de Variáveis:

- Variáveis como i, l, c, linha, coluna, jogador, ganhou, jogadas, numero são usadas para controlar o fluxo do jogo e armazenar valores temporários.
- Struct Jogador jogadores[2]: Array para armazenar informações dos dois jogadores.
- Char jogo[3][3]: Representa o tabuleiro do jogo da velha, uma matriz 3x3 de caracteres.

- Int loop = 1: Controla o loop principal do menu.

```
int main(int argc, char *argv[]) {

    int i, l, c, linha, coluna, jogador, ganhou, jogadas, numero = 0;
    struct Jogador jogadores[2];
    char jogo [3][3];
    int loop = 1;
```

2. Loop do Menu:

- O loop continua enquanto loop for igual a 1. Dentro do loop, o menu é exibido e a escolha do usuário é lida usando o número = lerNumero().

```
while(loop==1){ // menu do jogo
system("cls");
printf("\n\tJOGO DA VELHA\n\n");
printf("1 - JOGAR\n");
printf("2 - VER RANKING\n");
printf("3 - CREDITOS\n");
printf("4 - SAIR\n\n");
printf("Digite a opcao desejada: ");
numero = lerNumero();
```

3. Opções do Menu:

- Opção 1 (Jogar):

Solicita os nomes dos jogadores.
Inicializa variáveis de controle como jogador, ganhou e jogadas.
Inicializa o tabuleiro com espaços em branco.
Dentro de um loop do-while, o jogo continua até que alguém vença ou todas as células sejam preenchidas (empate).
Mostra o tabuleiro e lê as coordenadas de entrada dos jogadores para suas jogadas.
Verifica condições de vitória após cada jogada.
Se alguém ganhar, incrementa o número de vitórias desse jogador.

```
switch(numero){
    case 1:{
        printf("Digite o nome do jogador 1 (apenas 1 nome): ");
        scanf("%s", jogadores[0].nome);

        printf("\nDigite o nome do jogador 2 (apenas 1 nome): ");
        scanf("%s", jogadores[1].nome);

        jogador = 1;
        ganhou = 0;
        jogadas = 0;
        // inicializando a matriz
        for(l=0; l<3; l++){
            for(c=0; c<3; c++){
                jogo[l][c] = ' ';
            }
        }

        do{ // repetir atÃ© alguÃ©m ganhar ou dar velha
            // imprimir o jogo
            printf("\n\n\t 0   1   2\n\n");
            for(l=0; l<3; l++){
                for(c=0; c<3; c++){
                    if(c==0){
                        printf("\t");
                    }
                    printf(" %c ", jogo[l][c]);
                    if(c<2){
                        printf("|");
                    }
                    if(c==2){
```

(Consultar o apêndice para visualizar o restante do “Case 1”.)

- Opção 2 (Ver Ranking):

Exibe o número de vitórias de cada jogador.

```
case 2:{
    printf("\n\tRANKING\n\n");

    printf("Jogador 1: %d vitorias\n", jogadores[0].vitorias);
    printf("Jogador 2: %d vitorias\n", jogadores[1].vitorias);
    break;
}
```

- Opção 3 (Créditos):

Mostra informações sobre os desenvolvedores.

```
case 3:{
    printf("\n\t\t\t\tCREDITOS\n\n");

    printf("\n\tDesenvolvido por Anna Maria, Maria Eduarda e Milena Azevedo\n");
    printf("\n\tAlunos do UNIPE - Ciencia da Computacao\n");
    printf("\n\tTurma: Tecnicas e Desenvolvimento de Algoritmos - 1B 2023.2\n\n");
    break;
}
```

- Opção 4 (Sair):

Encerra o loop do menu, saindo do jogo.

```
case 4:{
    printf("\nSaindo do jogo...\n");
    loop = 0;
    break;
}
```

4. Controle de Jogo:

- O jogo alterna entre os dois jogadores, marcando 'o' ou 'x' no tabuleiro.
- Verifica as condições de vitória: linhas, colunas e diagonais completas.
- Anuncia o vencedor e atualiza o número de vitórias.
- Em caso de empate (todas as células preenchidas sem um vencedor), anuncia que o jogo acabou sem ganhador.

5. Retorno ao Menu ou Saída do Jogo:

- Após uma rodada de jogo ou visualização do ranking/créditos, o usuário pode escolher retornar ao menu principal ou sair do jogo.

```
if (loop != 0) {
    printf("\nDigite '1' para voltar ao menu, ou '0' para sair do jogo.\n");
    loop = lerNumero();
}
```

Apêndice

Código fonte:

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <locale.h>

/* Jogo da Velha. Projeto para Técnicas e Desenvolvimento de Algoritmos */

struct Jogador{
    char nome[50];
    int vitorias;
};

int lerNumero() { // Função para tratar erros do usuário
    setlocale (LC_ALL, "portuguese");
    int numero;
    while (scanf("%d", &numero) != 1) {
        printf("\n Opção Inválida! Digite um número: \n ");
        // limpa o buffer de entrada
        while (getchar() != '\n');
    }
    return numero;
}

int main(int argc, char *argv[]) {

    int i, l, c, linha, coluna, jogador, ganhou, jogadas, numero = 0;
    struct Jogador jogadores[2];
    char jogo [3][3];
    int loop = 1;

        while(loop==1){ // menu do jogo
        system("cls");
        printf("\n\tJOGO DA VELHA\n\n");
        printf("1 - JOGAR\n");
        printf("2 - VER RANKING\n");
        printf("3 - CRÉDITOS\n");
        printf("4 - SAIR\n\n");
        printf("Digite a opcao desejada: ");
        numero = lerNumero();

        switch(numero){
            case 1:{
                printf("Digite o nome do jogador 1 (apenas 1 nome): ");
                scanf("%s", jogadores[0].nome);

                printf("\nDigite o nome do jogador 2 (apenas 1 nome): ");
                scanf("%s", jogadores[1].nome);

                jogador = 1;
                ganhou = 0;
                jogadas = 0;
                // inicializando a matriz
                for(l=0; l<3; l++){
                    for(c=0; c<3; c++){
                        jogo[l][c] = ' ';
                    }
                }

                do{ // repetir atÃ© alguÃ©m ganhar ou dar velha
                    // imprimir o jogo
                    printf("\n\n\t 0   1   2\n\n");
                    for(l=0; l<3; l++){
                        for(c=0; c<3; c++){
                            if(c==0){
                                printf("\t");
                            }
                            printf(" %c ", jogo[l][c]);
                            if(c<2){
                                printf("|");
                            }
                            if(c==2){
                                printf("  %d", l);
                            }
                        }
                        if(l<2){
                            printf("\n\t-----------");
                        }
                        printf("\n");

                    }

                    // ler cordenadas
                    do{
                        printf("\nJOGADOR 1 (%s) = o\nJOGADOR 2 (%s) = x\n", jogadores[0].nome, jogadores[1].nome);
                        printf("\nJOGADOR %d: Digite a linha e a coluna que deseja jogar (observação: dê um espaco entre os números): ", jogador);
                        scanf("%d%d", &linha, &coluna);
                    }while(linha<0 || linha>2 || coluna<0 || coluna>2 || jogo[linha][coluna]!=' ');

                    // salvar coordenadas
                    if(jogador==1){
                        jogo[linha][coluna]='o';
                        jogador++;
                    }else{
                        jogo[linha][coluna]='x';
                        jogador = 1;
                    }
                    jogadas++;

```

```c
                    // ganhar por linha
                    if(jogo[0][0] == 'o' && jogo[0][1] == 'o' && jogo[0][2] == 'o' ||
                       jogo[1][0] == 'o' && jogo[1][1] == 'o' && jogo[1][2] == 'o' ||
                       jogo[2][0] == 'o' && jogo[2][1] == 'o' && jogo[2][2] == 'o'){
                        printf("\nO jogador 1 (%s) venceu!\n", jogadores[0].nome);
                        jogadores[0].vitorias++;
                        ganhou = 1;
                    }

                    if(jogo[0][0] == 'x' && jogo[0][1] == 'x' && jogo[0][2] == 'x' ||
                       jogo[1][0] == 'x' && jogo[1][1] == 'x' && jogo[1][2] == 'x' ||
                       jogo[2][0] == 'x' && jogo[2][1] == 'x' && jogo[2][2] == 'x'){
                        printf("\nO jogador 2 (%s) venceu!\n", jogadores[1].nome);
                        jogadores[1].vitorias++;
                        ganhou = 1;
                    }

                    // ganhar por coluna
                    if(jogo[0][0] == 'o' && jogo[1][0] == 'o' && jogo[2][0] == 'o' ||
                       jogo[0][1] == 'o' && jogo[1][1] == 'o' && jogo[2][1] == 'o' ||
                       jogo[0][2] == 'o' && jogo[1][2] == 'o' && jogo[2][2] == 'o'){
                        printf("\nO jogador 1 (%s) venceu!\n", jogadores[0].nome);
                        jogadores[0].vitorias++;
                        ganhou = 1;
                    }

                    if(jogo[0][0] == 'x' && jogo[1][0] == 'x' && jogo[2][0] == 'x' ||
                       jogo[0][1] == 'x' && jogo[1][1] == 'x' && jogo[2][1] == 'x' ||
                       jogo[0][2] == 'x' && jogo[1][2] == 'x' && jogo[2][2] == 'x'){
                        printf("\nO jogador 2 (%s) venceu!\n", jogadores[1].nome);
                        jogadores[1].vitorias++;
                        ganhou = 1;
                    }

                    // ganhar na diagonal principal
                    if(jogo[0][0] == 'o' && jogo[1][1] == 'o' && jogo[2][2] == 'o'){
                        printf("\nO jogador 1 (%s) venceu!\n", jogadores[0].nome);
                        jogadores[0].vitorias++;
                        ganhou = 1;
                    }
                    if(jogo[0][0] == 'x' && jogo[1][1] == 'x' && jogo[2][2] == 'x'){
                        printf("\nO jogador 2 (%s) venceu!\n", jogadores[1].nome);
                        jogadores[1].vitorias++;
                        ganhou = 1;
                    }

                    // ganhar na diagonal secundÃ¡ria
                    if(jogo[0][2] == 'o' && jogo[1][1] == 'o' && jogo[2][0] == 'o'){
                        printf("\nO jogador 1 (%s) venceu!\n", jogadores[0].nome);
                        jogadores[0].vitorias++;
                        ganhou = 1;
                    }
                    if(jogo[0][2] == 'x' && jogo[1][1] == 'x' && jogo[2][0] == 'x'){
                        printf("\nO jogador 2 (%s) venceu!\n", jogadores[1].nome);
                        jogadores[1].vitorias++;
                        ganhou = 1;
                    }
                }while(ganhou == 0 && jogadas < 9);

                if(ganhou == 0){
                    printf("\nO jogo finalizou sem ganhador!\n");
                }
                break;
            }
            case 2:{
                printf("\n\tRANKING\n\n");

                printf("Jogador 1: %d vitórias\n", jogadores[0].vitorias);
                printf("Jogador 2: %d vitórias\n", jogadores[1].vitorias);
                break;
            }
            case 3:{
                printf("\n\t\t\t\tCREDITOS\n\n");

                printf("\n\tDesenvolvido por Anna Maria, Maria Eduarda e Milena Azevedo\n");
                printf("\n\tAlunos do UNIPÊ - Ciência da Computação\n");
                printf("\n\tTurma: Técnicas e Desenvolvimento de Algoritmos - 1B 2023.2\n\n");
                break;
            }
            case 4:{
                printf("\nSaindo do jogo...\n");
                loop = 0;
                break;
            }
            default:{
                printf("\nOpção Inválida!\n");
                break;
            }
        }
        if (loop != 0) {
            printf("\nDigite '1' para voltar ao menu, ou '0' para sair do jogo.\n");
            loop = lerNumero();
        }
    }

    return 0;
}
```