

Curso: Ciência da Computação

Período: 1

Sala: C

Professor: Wallace Bonfim

Matéria: Técnicas e desenvolvimento do Algoritmo

Alunos: Emmanuel Lacerda Ribeiro (32853432), Erika Regina Pessoa Maciel (40081231), Isadora Garcez Alves (40658058) e Vitória Gonçalves da Silva (39882519)

RELATÓRIO – PROJETO DE TÉCNICAS E DESENVOLVIMENTO DE ALGORITMOS

Objetivo:

Este relatório descreve o desenvolvimento do projeto "Jogo da Velha", realizado pelos alunos da disciplina de Técnicas e Desenvolvimento de Algoritmos, com o objetivo de aplicar conceitos de programação e algoritmos na criação de um jogo simples.

Divisão de Tarefas:

Em 12 de novembro de 2024, formamos o grupo e decidimos que o projeto seria o desenvolvimento de um jogo da velha. As responsabilidades foram divididas da seguinte forma:

- Relatório: Erika Regina Pessoa Maciel
- Desenvolvimento do Código: Vitória Gonçalves da Silva e Emmanuel Lacerda Ribeiro
- Apresentação do Código: Isadora Garcez Alves

Cronograma e Atividades:

- **12/11/2024 - Formação do Grupo e Definição do Projeto:**

O grupo se reuniu para discutir o projeto. Após algumas ideias, decidimos desenvolver um jogo da velha. A divisão das tarefas foi feita e o trabalho começou a ser organizado.

- **13/11/2024 - Pesquisa e Testes:**

Vitória iniciou a pesquisa sobre a implementação do jogo, realizando testes e analisando códigos de outras fontes para encontrar a melhor abordagem.

- **19/11/2024 - Desenvolvimento do Código:**

A programação do jogo foi iniciada, incluindo a criação do menu e a implementação dos principais algoritmos. No dia 20/11/2024, conseguimos finalizar a primeira versão funcional do código.

- **21/11/2024 - Finalização e Ajustes:**

Finalizamos o código, acrescentamos comentários explicativos e fizemos os testes finais. Vitória também convidou todos os membros a colaborarem no repositório do projeto no GitHub.

- **25/11/2024 - Apresentação:**

A apresentação final do projeto será realizada no dia 25 de novembro de 2024, quando iremos demonstrar o funcionamento do código e compartilhar o que foi aprendido durante o processo.

Conclusão:

O projeto foi completado com sucesso, atendendo aos objetivos propostos. A divisão de tarefas foi eficiente, e o trabalho em equipe permitiu que o desenvolvimento do jogo da velha fosse realizado de maneira satisfatória. O código final foi disponibilizado no GitHub, e a apresentação final será feita conforme o cronograma.

SOBRE O JOGO:

Introdução:

Jogo da Velha (também conhecido como **Tic-Tac-Toe**) é um jogo clássico de dois jogadores que jogam alternadamente em um tabuleiro de 3x3 ou 5x5 (no caso deste código), com o objetivo de preencher uma linha, coluna ou diagonal com os mesmos símbolos. Os jogadores podem ser humanos ou, no caso de um jogador, o computador pode assumir o papel do oponente.

Regras do Jogo:

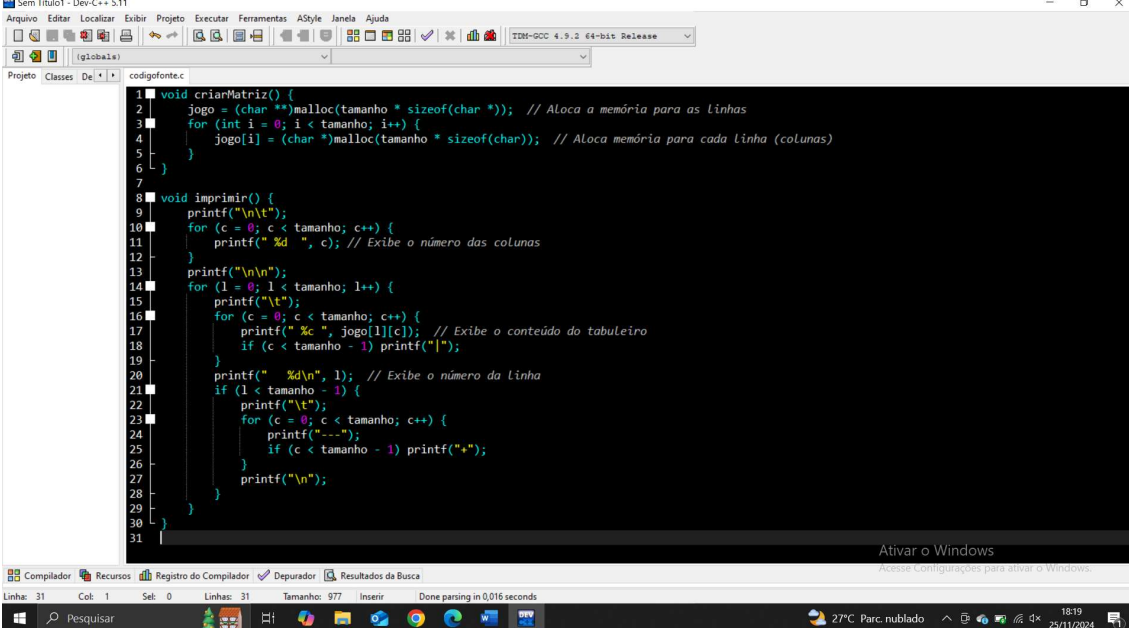
1. O tabuleiro tem 9 ou 25 espaços (dependendo do tamanho escolhido), inicialmente vazios.
2. O jogador 1 (X) começa a partida e faz sua jogada colocando o símbolo **X** em um dos espaços vazios.
3. O jogador 2 (O) ou o computador, dependendo da escolha, faz sua jogada colocando o símbolo **O** em um espaço vazio.
4. O objetivo é alinhar três (ou cinco) símbolos iguais em uma linha, coluna ou diagonal.
5. O jogo termina quando:
 - Um dos jogadores/alguém vence (alinha 3 ou 5 símbolos).
 - Ou se todas as posições do tabuleiro estiverem preenchidas sem vencedor, resultando em um empate.
6. Após cada jogo, o ranking é atualizado, contabilizando as vitórias de cada jogador.

Resultados: Descrição geral do jogo, com exemplificação de código fonte

O jogo foi implementado em **C** e oferece dois modos de jogo:

1. **Modo 1:** Jogar com 2 jogadores humanos (local).
2. **Modo 2:** Jogar contra o computador, onde o primeiro jogador é humano e o segundo jogador é controlado pelo computador.

Exemplo de código fonte:



```
1 void criarMatriz() {
2     jogo = (char **)malloc(tamanho * sizeof(char *)); // Aloca a memória para as linhas
3     for (int l = 0; l < tamanho; l++) {
4         jogo[l] = (char *)malloc(tamanho * sizeof(char)); // Aloca memória para cada linha (colunas)
5     }
6 }
7
8 void imprimir() {
9     printf("\n\t");
10    for (c = 0; c < tamanho; c++) {
11        printf(" %d ", c); // Exibe o número das colunas
12    }
13    printf("\n\n");
14    for (l = 0; l < tamanho; l++) {
15        printf("\t");
16        for (c = 0; c < tamanho; c++) {
17            printf(" %c ", jogo[l][c]); // Exibe o conteúdo do tabuleiro
18            if (c < tamanho - 1) printf("|");
19        }
20        printf(" %d\n", l); // Exibe o número da linha
21        if (l < tamanho - 1) {
22            printf("\t");
23            for (c = 0; c < tamanho; c++) {
24                printf("----");
25                if (c < tamanho - 1) printf("+");
26            }
27            printf("\n");
28        }
29    }
30 }
31 }
```

Dificuldades encontradas e soluções implementadas

Dificuldades:

1. Gerenciamento de memória dinâmica:

- Como o tamanho do tabuleiro pode ser alterado (3x3 ou 5x5), o uso correto da **memória dinâmica** foi crucial. Caso a memória não fosse alocada ou liberada corretamente, o jogo poderia travar ou apresentar erros.

Solução: Utilizamos malloc() para alocar a memória para o tabuleiro, e free() para liberar a memória quando o jogo termina. Isso evita vazamentos de memória. A alocação e liberação de memória são feitas nas funções criarMatriz() e liberarMatriz().

2. Lógica de verificação de vencedor:

- Checar se um jogador venceu ou se o jogo terminou foi um desafio, especialmente porque o tabuleiro pode ser de diferentes tamanhos (3x3 ou 5x5), o que torna a lógica mais complexa.

Solução: A função verificarVencedor() percorre todas as linhas, colunas e diagonais para verificar se algum jogador conseguiu alinhar os símbolos. Se encontrar uma linha, coluna ou diagonal com três ou cinco símbolos iguais, o jogo termina com um vencedor.

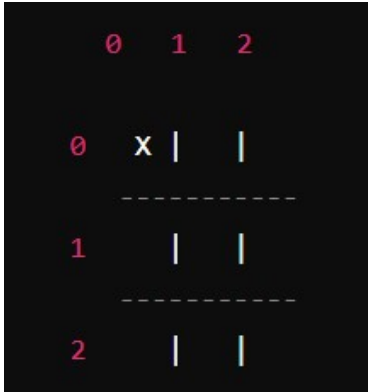
3. Implementação do modo computador:

- Programar o computador para jogar automaticamente foi um desafio, pois exigia uma lógica de tomada de decisão (ainda que simples), sem inteligência artificial avançada.

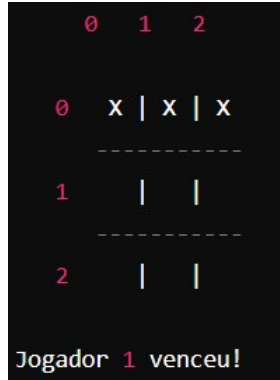
Solução: Para simplificar, o computador simplesmente escolhe o primeiro espaço vazio que encontra no tabuleiro. Isso mantém o jogo funcional e divertido, sem a complexidade de uma IA avançada.

Prints das telas:

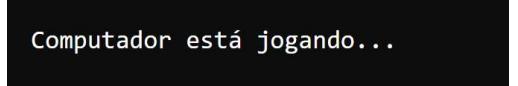
Tela 1:



Tela 2:



Tela 3:



Código completo no GitHub.