Católica
Universidade Católica de Brasília

Pró-Reitoria Acadêmica
Cursos de Ciência da Computação
e Análise e Desenvolvimento de Sistemas
Trabalho Disciplinar de Estrutura de Dados

ATIVIDADE AVALIATIVA: ESTRUTURA DE DADOS

Autores: Gabriel Victor Nasulicz Martins,
Gabriel Morais Resplandes,
Giovana Pacheco Velasco,
Guilherme Vaz Gomes,
Maria Luiza Barbosa Ferreira da Silva,
Raíssa Herculano de Matos

Orientador: Zoé Roberto Magalhães Junior

Brasília - DF
2025

GABRIEL VICTOR NASULICZ MARTINS - UC24200813

GIOVANA PACHECO VELASCO - UC24200330

GUILHERME VAZ GOMES – UC24100762

MARIA LUIZA BARBOSA FERREIRA DA SILVA - UC23200096

RAÍSSA HERCULANO DE MATOS - UC23200394

ATIVIDADE AVALIATIVA:

ESTRUTURA DE DADOS.

Documento apresentado aos Cursos de graduação de Ciência da Computação da Universidade Católica de Brasília, como requisito parcial para obtenção da aprovação na disciplina de Estrutura de Dados.

Orientador: Prof. Zoé Roberto Magalhães Junior

Brasília
2025

1. OBJETIVO DA SOLUÇÃO:

Este projeto tem como objetivo principal aplicar os conhecimentos da disciplina de Estrutura de Dados através do desenvolvimento de um jogo de cartas: o Blackjack (ou 21), usando a linguagem C. Durante o desenvolvimento, foram utilizados conceitos como listas encadeadas, ponteiros, alocação dinâmica de memória, uso de structs, enums, manipulação de arquivos e controle de fluxo com menus interativos.

O jogo é executado no terminal, permitindo que o jogador jogue contra o computador (dealer), seguindo as regras oficiais do Blackjack. Ao final de cada rodada, as pontuações são salvas em um arquivo de texto chamado 'placar.txt'.

2. DESCRIÇÃO DA ENTRADA E SAÍDA DE DADOS:

O jogo interage com o usuário por meio do terminal. Logo no início, o usuário informa seu nome. Em seguida, um menu é apresentado com as opções:

- Iniciar um novo jogo;

- Ver o histórico de pontuação (placar);

- Ler as regras do jogo;

- Sair do programa.

Durante a partida, o jogador decide se deseja comprar mais cartas ou parar. O dealer joga automaticamente.

Como saída, o jogo exibe as cartas tiradas, a pontuação atual e o resultado (vitória ou derrota). Todos os resultados são armazenados no arquivo 'placar.txt'.

3. ALGORITMO DE ORDENAÇÃO UTILIZADO:

O projeto faz uso de várias estruturas importantes da linguagem C:

- Listas encadeadas: para representar o baralho e a mão de cada jogador.

- Structs: para modelar as cartas (struct Carta) e os jogadores (struct Jogador).

- Enums: para representar os naipes (Copas, Espadas, etc.) e os valores das cartas (Ás, 2... Rei).

- Arquivos: leitura e escrita no placar usando funções como fopen, fprintf e fgets.

- Geração de números aleatórios: srand() e rand() para embaralhar as cartas.

- Modularização do código: o programa está dividido em arquivos .c e .h, organizando funções de forma clara.

4. ESTRUTURA DO CÓDIGO E PRINCIPAIS FUNÇÕES IMPLEMENTADAS:

O programa está dividido em três arquivos principais:

- main.c: contém a função principal e o menu do jogo.

- blackjack.c: implementa toda a lógica do jogo, como criação do baralho, compra de cartas e cálculo de pontuação.

- blackjack.h: arquivo de cabeçalho com as definições de structs, enums e funções.

Algumas funções importantes:

- inicializarBaralho(): cria as 52 cartas do jogo.

- embaralharBaralho(): embaralha as cartas de forma aleatória.

- comprarCarta(): remove a carta do topo do baralho e entrega ao jogador.

- calcularPontuacao(): soma os valores das cartas, tratando o Ás como 11 ou 1 dependendo do caso.

- turnoDoJogador() / turnoDoDealer(): controlam os turnos do jogador e da banca.

- salvarPontuacao(): registra o nome e a pontuação do jogador em um arquivo.

- exibirPlacar(): exibe o conteúdo salvo no placar.

6. PROBLEMAS ENCONTRADOS E SOLUÇÕES ADOTADAS:

- Manipulação de ponteiros e memória: Como as cartas são representadas com listas encadeadas, era necessário usar malloc() para alocar e free() para liberar a memória. Criamos a função liberarMao() para evitar vazamentos.

- Pontuação com Ás: O Ás pode valer 1 ou 11. Para resolver isso, a função calcularPontuacao() analisa a mão do jogador e ajusta o valor para não ultrapassar 21 pontos.

- Validação da entrada: Para evitar que o usuário digitasse letras ou caracteres inválidos no menu, criamos uma função chamada limpar_stdin() que limpa o buffer de entrada quando necessário.

- Organização do código: A divisão entre .c e .h ajudou a manter o projeto modular, separando bem as funções e a lógica do jogo.

5. CONCLUSÃO:

O desenvolvimento do jogo Blackjack foi uma experiência prática muito rica, onde colocamos em prática conceitos importantes aprendidos em sala. A implementação com listas encadeadas e ponteiros nos fez entender melhor como funciona a alocação de memória e manipulação de dados.

Além disso, usamos funções para modularizar o código, criamos structs bem organizadas e aprendemos a trabalhar com arquivos para armazenar informações do jogo. O código ficou bem dividido, de fácil leitura e manutenção.

Como melhorias futuras, podemos incluir múltiplos jogadores, implementar uma interface gráfica e melhorar a experiência de usuário com mensagens e feedback mais amigáveis.