SCC0502 Algoritmos e Estruturas de Dados 1

Exercício 3 - 21!!!!!

Objetivo

O objetivo deste exercício prático é estimular os estudantes a se familiarizarem com o comportamento e a lógica associados à estrutura de dados **pilha.**

Queremos que os alunos se habituem com a implementação desssa estrutura e consigam entender as funções básicas, que garantem a execução adequada desse *TAD*, em diferentes contextos.

Para acostumar os alunos com conceitos de modularização e boas práticas de escrita de código, será exigido o uso de múltiplos arquivos *.c* e *.h* no projeto, bem como a construção de um arquivo **Makefile**, responsável por gerenciar a execução do programa.

Descrição

Sendo alunos da USP, somos fãs de jogos de cartas como truco, pôquer ...

Dentre esses jogos temos o *BlackJack*, famigerado 21. A dinâmica mais básica para o jogo consiste no apostador escolher cartas da pilha do baralho e somar seus valores até atingir a conta de 21, e nesse caso ganhar, ou ultrapassar e perder.

Espera-se que seu programa seja capaz atender aos seguintes requisitos:

1. Ler um conjunto de 52 pares de valores (nipe, símbolo), representando as cartas, previamente embaralhadas da sua pilha.
2. Retirar cartas do topo da pilha e fazer o cálculo da soma.
3. Indicar se o apostador teve sorte e ganhou, ou estava em um mau dia e perdeu.

Entrada

Seu algoritmo deve receber as 52 cartas do baralho no formato: "Nipe Símbolo".

Nipes = {"Espadas", "Paus", "Ouros", "Copas"}.

Símbolos = {'1', '2', ..., 10, 'V', 'D', 'K' }.

Exemplo de entrada:

```
Ouros 1
Ouros 7
Espadas 5
Espadas 9
Paus 9
Paus V
Copas V
Ouros 10
Copas 3
Espadas 3
Paus 6
Ouros 9
Paus D
Copas 9
Paus 10
Espadas 6
Copas R
Espadas 2
Ouros V
Paus 8
Espadas 4
Paus 7
Ouros 2
Ouros D
Copas 2
Copas 5
Copas 6
Ouros 8
Copas 10
Ouros R
Espadas 10
Paus 4
Copas 1
Espadas D
Paus R
Ouros 5
Copas D
Espadas 7
Paus 1
Espadas R
Ouros 3
Ouros 4
Copas 4
Copas 8
Espadas 8
Ouros 6
Paus 5
Paus 2
Espadas 1
Espadas V
Copas 7
Paus 3
```

Saída

Como saída, seu programa deve indicar o resultado do jogo realizado.

Saída da caso acima:

```
Ganhou ;)
```

Caso o jogador perca:

```
Perdeu :(
Soma :: valor
```

Observações da implementação

Como é descrito na sessão objetivos desse documento, não queremos apenas que os alunos resolvam o problema, mas que utilizem métodos que serão comuns no decorrer da disciplina.

Devido a esse objetivo, será exigido que vocês desenvolvam seu projeto representando o objeto **carta** como struct e com a implementação do **TAD pilha** para construir o baralho de forma abstrata. A memória deve ser alocada **dinamicamente** e ser devidamente liberada ao fim da execução.

Utilizar múltiplos arquivos *.c* e *.h* para separar a implementação e a responsabilidade dos métodos de cada objeto.

Construir funções para realizar operações repetitivas, ou seja, modularizar adequadamente seu código.

Escrever um arquivo **Makefile** que será responsável por gerenciar a execução do projeto dentro da plataforma *run.codes*.

Observações da avaliação

A avaliação do seu programa será feita além do resultado da plataforma *run.codes*. Portanto, ter um bom resultado com os casos de teste, não será suficiente para garantir a **nota máxima** e nem a **aprovação do exercício**.

Caso seu projeto não satisfaça os pontos exigidos nos **objetivos** e explicitados nas **observações de implementação**, sua nota poderá ser reduzida ou ser desconsiderada.

Cópias de código entre alunos, acusadas pela plataforma, resultarão imediatamente em **zero** aos dois ou mais alunos envolvidos.