Desafio de Estrutura de Dados "FreeCell"

Universidade Tecnológica Federal do Paraná (UTFPR), campus Apucarana
Curso de Engenharia de Computação
Disciplina de Estrutura de Dados - EDCO3A
Prof. Dr. Rafael Gomes Mantovani

---

1 Objetivo

O presente desafio é baseado no desafio 1 do livro Estruturas de Dados com Jogos, de Roberto Ferrari *et. al.* [1]. O objetivo é implementar uma versão do jogo `FreeCell` (sem interface gráfica) usando estruturas de dados.

---

2 Descrição

O `FreeCell`[1] é um jogo de paciência, que pode ser jogado também com um baralho de verdade e não apenas no mundo virtual. O `FreeCell` ficou muito conhecido depois que passou a ser distribuído junto com o sistema operacional Windows (Figura 1). Inicialmente, as 52 cartas de um baralho são distribuídas aleatoriamente em pilhas de cartas, que ficam na parte de baixo da tela. Essas pilhas são chamadas de *Pilhas intermediárias*. O objetivo é empilhar todas as 52 cartas nas pilhas que ficam no canto superior direito da tela, chamadas de *Pilhas Definitivas*. Mas, nas *Pilhas Definitivas* é preciso empilhar as cartas em sequência (A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K) e organizadas por naipes (copas, espadas, paus e ouros). Há uma *Pilha Definitiva* para cada naipe.

A movimentação das cartas precisa seguir algumas regras:

- Só é possível retirar a carta que está no topo de uma *Pilha Intermediária*;
- É possível passar cartas de uma *Pilha Intermediária* para outra, mas inserindo as cartas sempre no topo da pilha, em ordem decrescente, e em cores alternadas - pretas sobre vermelhas, e vermelhas sobre pretas.

Existem ainda quatro *Espaços para Movimentação de Cartas*, no canto superior esquerdo da tela. Nesses espaços é possível colocar qualquer carta, temporariamente. O uso desses espaços pode ajudar bastante no andamento do jogo.

---

[1] https://freecell.eco.br/paciencia-freecell/

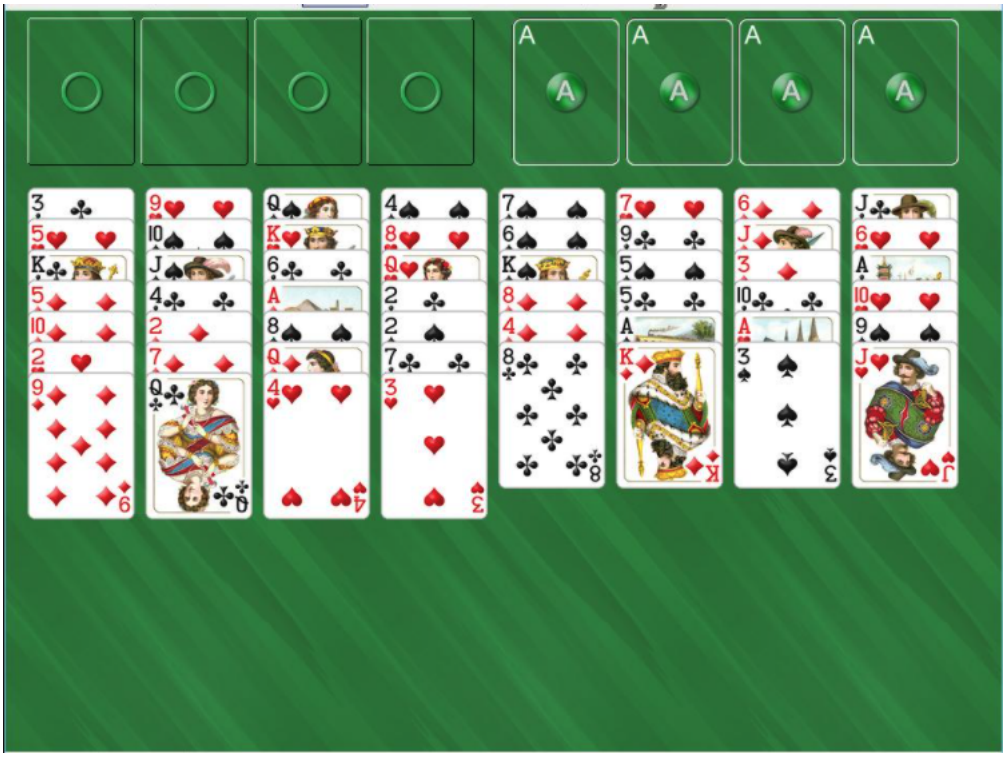

Figura 1: Tela de jogo do `FreeCell`. Na imagem é possível ver a existência de 4 espaços auxiliares na parte superior à esquerda; 4 pilhas definitivas na parte superior à direita; e 8 pilhas intermediárias, com as cartas embaralhadas na parte inferior.

3 O que fazer?

Implemente um jogo de `FreeCell` usando tipos abstratos de dados. O jogo não necessita de interface gráfica, gere saídas impressas pelo console, e controle as movimentações do jogo por meio de ações do usuário realizadas pelo teclado. Algumas perguntas que podem ajudar na modelagem do problema:

- Quais estruturas seriam necessárias para armazenar os valores dos naipes e cartas?
- Qual seria a melhor estratégia para armazenar e manipular as informações referentes a essas pilhas de cartas dentro do seu programa? Defina o(s) tipo(s) abstrato(s) de dados para que se possa implementar e manipular suas diferentes pilhas dentro do jogo, e especifique bem quais tipos seriam usados para implementar quais pilhas (intermediárias e definitivas).
- Além disso, como você armazenaria a sequência de cartas em cada uma das pilhas (definitivas e intermediárias) e implementaria as regras que restringem a movimentação das cartas? Nesse sentido, defina duas funções 'inteligentes' para manipular as cartas das de cartas em uma *Pilha Intermediária*, e outra para inserir cartas em uma *Pilha Definitiva*. **Dica**: Nas pilhas definitivas só podem ser armazenadas cartas de um mesmo naipe, e valores inseridos de forma crescente. Já nas pilhas intermediárias, as inserções

são feitas estabelecendo a ordem decrescente dos símbolos das cartas, e alternando as cores dos naipes (preta sobre vermelha, e vermelha sobre preta). Se necessário, acesse o jogo pelo link no rodapé da página anterior, e jogue algumas partidas para entender a lógica e regras do jogo.

---

4 O que deve ser entregue?

Elaborar um relatório técnico onde conste:

a) Listagem dos programas em C;

b) Listagem dos testes ou simulações executadas;

c) Descrição sucinta (desenhos, diagramas) das estruturas de dados e as decisões tomadas relativas aos casos e detalhes de especificação que porventura estejam omissos no enunciado;

---

5 Orientações gerais

- Implementar também o controle de erros, para lidar com exceções que possam ocorrer ao longo da execução do programa;
- Para acompanhamento do desenvolvimento, criar um repositório individual com o código desenvolvido no `github`;
- Entrega do programa final: via Moodle. O aluno deve submeter o código-fonte, e relatório com as análises na página da disciplina no Moodle.
- **Data máxima para entrega**: fim do semestre letivo corrente;
- Os códigos desenvolvidos por cada aluno serão também verificados por ferramentas de plágio. Códigos iguais/similares terão nota zero.

---

6 Links úteis

Números aleatórios em C:

- http://linguagemc.com.br/valores-aleatorios-em-c-com-a-funcao-rand/
- https://www.ime.usp.br/~pf/algoritmos/aulas/random.html

- https://www.cprogressivo.net/2013/03/Como-gerar-numeros-aleatorios-em-C-com-a-rand-srand-e-seed.html
- http://www.cplusplus.com/reference/cstdlib/rand/

Referências

[1] Roberto Ferrari; Marcela Xavier Ribeiro; Rafael Loosli Dias; Maurício Falvo. Estruturas de dados com jogos. Elsevier, 2014.

[2] Thomas H. Cormen,; Ronald Rivest; Charles E. Leiserson; Clifford Stein. Algoritmos - Teoria e Prática - 3ª Ed. Elsevier - Campus, 2012.

[3] Nivio Ziviani. Projeto de algoritmos com implementações: em Pascal e C. Pioneira, 1999.

[4] Adam Drozdek. Estrutura De Dados E Algoritmos Em C++. Cengage, 2010.