FATEC JAHU

ESTRUTURA DE DADOS – EXERCÍCIOS DE REVISÃO E FIXAÇÃO

Prof. Me. Tiago Antonio da Silva

PILHA

1) Um mágico embaralha 5 cartas com os valores A, B, C, D, E e as empilha. Peça aos alunos para simular o processo de colocar as cartas na pilha (push) e depois revelar a ordem em que saem (pop), mostrando que o último a entrar é o primeiro a sair (LIFO).
   Desafio extra: inverter a ordem da pilha usando uma segunda pilha.

2) Simule as funcionalidades de "Desfazer" e "Refazer" de um editor de texto com duas pilhas: uma para as ações feitas, outra para as ações desfeitas.
   Exemplo: Digitar "A", "B", "C", desfazer duas vezes, refazer uma vez.

3) Módulos espaciais pousam um sobre o outro. O último a pousar é o primeiro a sair da base. Simule essa sequência com uma pilha e peça para exibir a ordem de retorno à nave.

FILA

4) Crianças entram na fila da montanha-russa. Cada criança leva x segundos para brincar. Simule a entrada (enqueue) e atendimento (dequeue), imprimindo quem está sendo atendido e o tempo restante.

5) Clientes fazem pedidos num drive-thru. Crie uma fila que registra os pedidos por nome e imprima a sequência de atendimento.
   Extra: calcule tempo total de espera se cada atendimento dura 2 minutos.

6) Implemente uma variação da fila onde clientes com mais de 60 anos passam na frente. (Dica: fazer com duas filas e intercalar atendimento.)

LinkedList

7) Cada nó representa um corredor. Simule a passagem do bastão com `insertAtEnd` e remova um corredor machucado com `removeByValue`. Mostre a lista após cada modificação.

8) Usuário monta uma lista de compras com `insertAtBeginning` e `insertAtEnd`. Permita remover itens com `removeByValue` e buscar com `find`. Imprima a lista em cada etapa.

9) Cada nó é uma música de uma playlist. Insira músicas no fim, remova uma específica e permita busca por nome.
   Desafio extra: Adicionar opção de "tocar próxima" (avançar no ponteiro).

DoublyLinkedList

10) Cada vagão é um nó. Simule adicionar vagões à frente e atrás. Mostre a composição do trem indo e voltando com `traverse` e `traverseReverse`.

11) Cada letra digitada é um nó. O cursor pode ir para frente e para trás (simulando o `next` e `prev`). Permita inserções e remoções em qualquer posição (`insertAt`, `removeAt`).

12) Use a estrutura para simular o histórico de páginas acessadas. Com `traverseReverse`, mostre a navegação para trás (voltar página).
   Extra: implementar "ir para página específica" usando find.