Universidade Federal do Maranhão
Departamento de Informática
Estrutura de Dados I
Prof. Anselmo Paiva

Lista de Exercícios 1 – Pilhas, Filas e TAD

1. Use as operações *push, pop, top* e *empty* para construir operações que façam o seguinte:
   a. Definir *i* com o segundo elemento a partir do topo da pilha, deixando a pilha sem seus dois elementos superiores.
   b. Definir *i* com o segundo elemento a partir do topo da pilha, deixando a pilha inalterada.
   c. Dado um inteiro *n,* definir *i* como o enésimo elemento a partir do topo da pilha, deixando a pilha sem seus *n* elementos superiores.
   d. Dado um inteiro *n,* definir *i* como o enésimo elemento a partir do topo da pilha, deixando a pilha inalterada.
   e. Definir *i* como o último elemento da pilha, deixando a pilha vazia.
   f. Definir *i* como o último elemento da pilha, deixando a pilha inalterada. (Dica: use outra pilha auxiliar.)
   g. Definir *i* como o terceiro elemento a partir do final da pilha.
2. Escreva um algoritmo para determinar se uma string de caracteres de entrada é da forma: *xCy,* onde *x* é uma string consistindo nas letras 'A' e 'B', e y é o inverso de *x* (isto é, se *x* = "ABABBA", *y* deve equivaler a "ABBABA"). Em cada ponto, você só poderá ler o próximo caractere da string.
3. Escreva um algoritmo para determinar se uma string de caracteres de entrada é da forma: a D b D c D...D z onde cada string, a, 6, ..., *z,* é da forma da string definida no Exercício 3 (Por conseguinte, uma string estará no formato correto se consistir em qualquer número de strings desse tipo, separadas pelo caractere 'D'.) Em cada ponto, você só poderá ler o próximo caractere da string.
4. Elabore um algoritmo que não use uma pilha para ler uma seqüência de operações *push e pop,* e determine se está ocorrendo underflow ou não em alguma operação *pop.*
5. Uma pilha, como um tipo de dado abstrato, pode armazenar de 0 a infinitos elementos. Porém, uma implementação de pilha prevê não mais que um certo número N de elementos. Explique o porquê desse limite, considerando algumas formas de se implementar uma pilha.
6. Seja uma estrutura de dados chamada de Deque (double ended queue), isto é, uma estrutura com duas extremidades, que permite inserção e remoção de elementos em ambas as extremidades.
   a) Defina essa estrutura de forma abstrata, isto é, os dados que podem ser armazenados e as operações (5 operações) que podem ser realizadas sobre esses dados. Explique o funcionamento da interface e os parâmetros utilizados.
   b) Defina uma estrutura de dados utilizando vetor que implemente o Deque. Justifique.
7. Implemente uma fila usando duas pilhas.

8. Faça um procedimento recursivo para procurar por um valor x em uma pilha de inteiros, ambos passados como parâmetros, sendo que, ao final, a pilha deverá permanecer intacta.
9. Implemente uma pilha dupla, assim chamada por manter duas pilhas (dois topos) compartilhando um mesmo vetor, com economia de memória. Uma pilha dupla possui, dois push´s, dois pop´s e assim por diante
10. Faça um procedimento RemoveElemento(int fila Q, int x) que elimina um certo x de uma fila Q sem alterar a ordem dos demais elementos.
11. Seja uma seqüência de E's e D's que significam ações de empilhar e desempilhar, respectivamente, elementos em/de uma certa pilha S, faça um algoritmo que verifique uma seqüência qualquer e retorne OK ou NOK para o caso de seqüência bem formada ou mal formada.
    Exemplo: EEEEEDD (bem formada); EDEDEEDDDEEE (mal formada).
12. Faça um procedimento interativo: PESQFILA(int fila Q, int x) que pesquisa em uma fila Q por um argumento x, ambos passados como parâmetro.O procedimento deve retornar V ou F caso encontre ou não o argumento. A fila, ao final do processo, não deve estar alterada, por isso deve ser utilizada uma estrutura auxiliar (pilha ou fila) para efetuar a pesquisa.
13. Elabore um método para manter duas pilhas dentro de um único vetor linear *$[spacesize]* de modo que nenhuma das pilhas incorra em estouro até que toda a memória seja usada, e uma pilha inteira nunca seja deslocada para outro local dentro do vetor. Escreva rotinas em C, *push1, push2, pop1* e *pop2,* para manipular as duas pilhas. *(Dica:* as duas pilhas crescem na direção da outra.)
14. Escreva uma função que receba duas filas P1 e P2, de inteiros, e seja capaz de remover suas repetições entre si, mantendo a mesma sequencia de atendimento da fila.
15. Escreva a função BUSCA_E_REMOVE(pilha p, valor v) que buscar o valor v numa pilha p e remove esse elemento da pilha usando apenas as operações de pop e push.
16. Dado uma pilha P qualquer, transcreva seus elementos para uma Fila f de tal maneira que o último elemento da pilha, deve ser o primeiro da fila e assim em diante.
17. Dado um TAD genérico, implementar a função REMOVE_BLOCO que recebe o tad e dois índices: início e fim. Todos os elementos dentro do inicio e fim (se existirem) devem ser removidos do TAD.