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

Lista de Exercícios 1 – Pilhas

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 certa linguagem de programação que traz embutido o tipo PILHA e o tipo FILA. Assim, pode-se fazer as seguintes construções:

```
INT PILHA X;
INT FILA Y;
INT A;
Y=22;
Y=222;
Y=2222;
X=Y;
```

```
X=Y;
A=X;
PRINT (A);
```

Observe a sequência de instruções e responda qual o valor de A impresso no vídeo ?

7. 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.
8. Faça um procedimento recursivo float Avalia_Prefixa (string exp_prefixa) que recebe uma string representado uma expressão em notação prefixa, onde cada operando possui apenas um dígito, e retorna um número real com o resultado da avaliação da expressão
9. Utilizando uma linguagem de programação ou pseudo-código descreva:
   a) uma estrutura que implemente uma fila de prioridades utilizando múltiplas filas e um procedimento job Recupera(filap Q) que recupera a próxima tarefa (job de maior prioridade) a ser executada.
   b) um procedimento int Armazena(filap Q, Job J) que armazena uma tarefa, segundo a prioridade J.prior.
   c) um procedimento int SemJob(filap Q) que retorna verdadeiro somente se não houver nenhuma tarefa de qualquer prioridade (fila de prioridades vazia)
   d) um procedimento int Insere (fila Q , job J) para inserir uma tarefa em uma fila de prioridades implementada usando uma única FILA comum. Observe a necessidade de classificar as TAREFAS segundo suas prioridades (menor o valor, maior a prioridade) nessa fila, para isso talvez seja necessário o uso de uma fila auxiliar.
10. Implemente uma fila usando duas pilhas.
11. 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.
12. 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
13. Seja a operação de potenciação (P) definida no conjunto dos números naturais a b = a * a * ... *a (b-vêzes). Explique como redefinir P recursivamente e mostre uma implementação para P recursivo.
14. 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.
15. 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).

16. Implemente uma matriz de inteiros bidimensional M[1:MAXLIN, 1:MAXCOL] utilizando um vetor de inteiros V [1: MAXLIN*MAXCOL]. Para isso defina as operacoes basicas :

    SAVE (vetor V, int i, int j , int x)
    Guarda um valor x nas coordenadas (i,j) da matriz implementada em V
    int GET (vetor V, int i , int j)
    Retorna o valor armazenado nas coordenadas (i,j) da matriz.
17. Faça um procedimento recursivo: PESQPILHA(int pilha S, int x) que pesquisa em uma pilha S por um argumento x, ambos passados como parâmetro.O procedimento deve retornar V ou F caso encontre ou não o argumento. A pilha, ao final do processo, não deve estar alterada.
18. 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.
19. Elabore um método para manter duas pilhas dentro de um único vetor linear *$[spacesize]* de modo que nenhuma das pilhas incorra em estouro até que toda a memória seja usada, e uma pilha inteira nunca seja deslocada para outro local dentro do vetor. Escreva rotinas em C, *push1*, *push2*, *pop1* e *pop2*, para manipular as duas pilhas. *(Dica:* as duas pilhas crescem na direção da outra.)
20. Transforme cada uma das seguintes expressões em prefixas e posfixas:
    a. *A + B - C*
    b. (A + B)*(C -*D)$E* F*
    c. (A + *B)*(C $(D-E) + F)-G*
    d. (A + (((B - *C)*(D -E)+F) I G)$(H - J)*
    a) A^(B^(C*(D/(E-F))))+G
    b) (A+B) * ((C+D) * (E + F)) ^ ((G+H) * (I + J)) * (K + L)
    e. (A ^ B) ^ (C ^ D)
21. Transforme cada uma das seguintes expressões prefixas em infixas:
    a. + - *ABC*
    b. +*A-BC*
    c. + + A - * $ *BCD/ + EF * GHI*
    d. + - $ *ABC * D ** EFG*
22. Transforme cada uma das seguintes expressões posfixas em infixas:
    a. *AB + C* -
    b. *ABC* + -
    c. *AB -C + DEF* - + $
    d. *ABCDE* - + $ * *EF* * -
23. Aplique o algoritmo de avaliação apresentado em aula para avaliar as seguintes expressões posfixas. Pressuponha que A = 1, B = 2, C = 3.
    a. *AB + C - BA + C* $ -
    b. *ABC* + * *CBA* - + *
24. Modifique a rotina eval de modo a aceitar como entrada uma string de caracteres de operadores e operandos representando uma expressão posfixa e criar a forma infixa

totalmente com parênteses. Por exemplo, AB + seria transformada em (A + B) e AB + C - seria transformada em ((A + *B)* - C).

25. Escreva um único programa combinando os recursos de eval e postfix para avaliar uma string infixa. Use duas pilhas, uma para operandos e outra para operadores. Não converta primeiramente a string infixa em posfixa e, em seguida, avalie a string posfixa, mas, em vez disso, avalie no decorrer da operação.
26. Escreva uma rotina prefix para aceitar uma string infixa e criar a forma prefixa dessa string, presumindo que a string seja lida da direita para a esquerda e a string prefixa seja criada da direita para a esquerda.
27. Escreva um programa em C para converter:
    a. uma string prefixa em posfixa
    b. uma string posfixa em prefixa
    e. uma string prefixa em infixa
    c. uma string posfixa em infixa
28. Escreva uma rotina em C, reduce, que aceite uma string infixa e forme uma string infixa equivalente com todos os parênteses supérfluos removidos. Isso pode ser feito sem usar uma pilha?