SCC502 – Algoritmos e Estruturas de Dados I

Prof.ª Maria Cristina Ferreira de Oliveira

2º sem. 2014

Trabalho 2 – Pilha e Recursividade

Prazo de entrega: 10/10/2014 no Run.Codes

(Trabalho Individual)

Introdução

O objetivo deste trabalho é, utilizando apenas as **operações sobre o TAD Pilha de inteiros** (alocar, desalocar, push, pop e numeroElementos) e **recursividade**, escrever funções para inserir e remover elementos em qualquer posição da pilha. Ou seja, a pilha será usada para ´simular´ operações de inserção e remoção em vetor.

Neste trabalho, você deve:

1. Definir um TAD pilha armazenado em um vetor alocado dinamicamente;
    a. Alocar a memória da pilha (criar);
    b. Desalocar a memória da pilha (destruir);
    c. Inserir um elemento no topo da pilha (push);
    d. Remover um elemento do topo da pilha (pop);
    e. Retornar o número de elementos da pilha (numeroElementos);
2. Escrever funções para:
    a. Inserir um elemento na i-ésima posição da pilha (**utilizando recursividade**);
    b. Remover um elemento da i-ésima posição da pilha (**utilizando recursividade**);
3. Considere que os casos de teste não farão operações ilegais na pilha (remover elementos de pilha vazia, por exemplo).

Atenção

Não utilize loops (while, for) nas funções para manipulação da pilha. Utilize funções recursivas, quando necessário.

Entrada

Os dados – fornecidos na entrada-padrão – contém apenas um caso de teste, que consiste em comandos para o sistema:

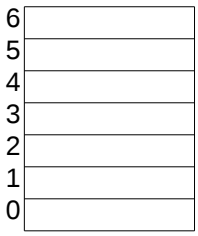
- “p <n>”: **push** – insere o número <n> no topo da pilha;
- “q”: **pop –** remove o número do topo da pilha e apresenta o valor na tela;
- “i <pos> <n>” - insere o número <n> na posição <pos> **da pilha** (começando do zero, a partir da base da pilha). Completa a pilha com zeros até chegar na posição desejada.
- “r <pos>” - remove o número <n> da posição <pos> **da pilha** e apresenta o valor na tela;
- “x” - sair do programa.

Exemplo:

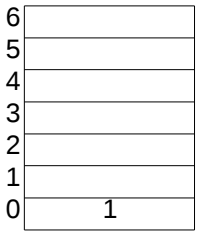
```
p 1
p 2
i 3 4
q
q
i 1 5
i 1 6
r 0
r 1
q
q
x
```

Execução do exemplo:

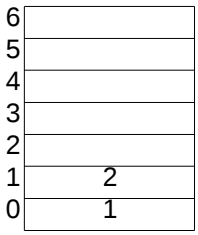
Pilha vazia



p 1

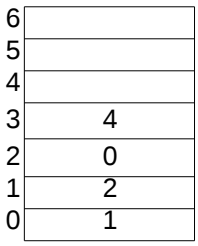


p 2



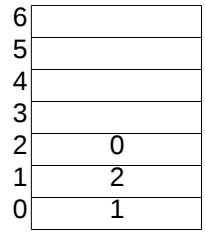
i 3 4

(este comando insere o valor 0 para completar a pilha)



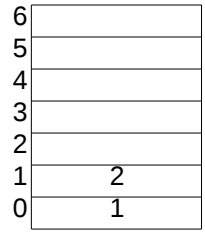
2 Pushs

q



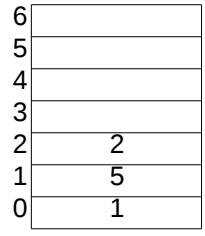
Retorna o valor 4

q



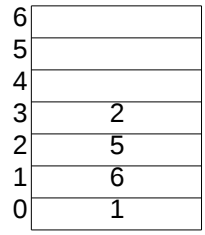
Retorna o valor 0

i 1 5



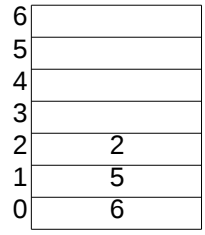
2 Pushs

i 1 6



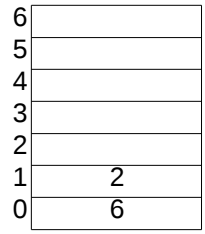
3 Pushs

r 0



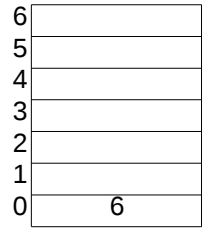
Retorna o valor 1, 3 Pushs

r 1



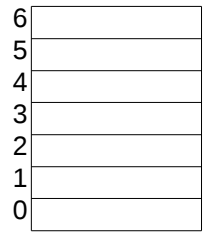
Retorna o valor 5, 1 Push

q



Retorna o valor 2

q



Retorna o valor 6

Saída

A saída - apresentada do dispositivo padrão – consiste de várias linhas, de acordo com o comando na entrada

- “p <n>”: push – Não apresenta nada na tela (nem quebras de linha);

- “q”: pop – remove o número do topo da pilha e apresenta o valor na tela [[printf(“%d\n”, valor);]];

- “i <pos> <n>” - Apresenta o número de operações push executadas para inserir o valor nessa posição [[printf(“%d\n”,numeroDePushs”);]];

- “r <pos>” - remove o número <n> da posição <pos> da pilha e apresenta o valor na tela e o número de operações push executadas para fazer isso [[printf(“%d %d\n”,valor, numeroDePushs);]];

- “x” - desalocar a pilha e sair do programa**.**

Para a entrada apresentada no exemplo, a saída esperada é:

```
2
4
0
2
3
1 3
5 1
2
6
```

Outras Informações Importantes

- O trabalho deve ser feito individualmente.
- O programa pode ser feito na linguagem C.
- Todas as submissões são checadas para evitar cópia/plágio/etc. Então, evite problemas e implemente o seu próprio código.
- Comente o seu código com uma explicação rápida do que cada função, método ou trecho importante de código faz (ou deveria fazer). Os comentários e a modularização do código (“.c”, “.h”, “Makefile”) serão checados e valem nota.
- Entradas/saídas devem ser lidas/escritas a partir dos dispositivos padrão, ou seja, use as funções “*printf(...)”* e “*scanf(...)”*. Para testar, arquivos podem ser redirecionados para/de seu programa na linha de comando utilizando os operadores < e >.
- Exemplo:

```
# ./trab2 < entrada.txt > saida.txt
```