Pilhas Dinâmicas – Lista 3 Estrutura de Dados
Prof. Leandro Colevati FATEC ZL

1. Considere o vetor a seguir:

0	5	7	-4	3	5	-2	-1	10	4	3	-6	2	-9	1	-5

Fazer um algoritmo, em Java, utilizando a biblioteca PilhaInt, inicializar uma pilha de inteiros e resolva conforme as condições:

a. O vetor deve ser percorrido do primeiro ao último elemento
b. Caso seja um número positivo ou 0, fazer o push do valor
c. Caso seja um número negativo, fazer o pop de 2 valores, somá-los, fazer o push do número negativo e o push do resultado da soma
d. Ao término do vetor, apresentar a quantidade de valores presentes na pilha.

2. Considere a pilha abaixo, já formada.

J
G
R
B
H
L
W

Demonstrando em código ou teste de mesa, criar uma sequência de operações de pilha que dê a seguinte saída:

Console:
R W

Pilha Final:

K
G
M
B
L

3. Considerando a biblioteca PilhaString, já criada, faça:
Criar um projeto Java (Palindromos) e importe a biblioteca PilhaStrings. Esse novo projeto irá receber uma cadeia de Strings do usuário e demonstrar para ele se essa palavra se trata ou não de um Palíndromo.
Palíndromo é uma cadeia de caracteres que se lê da direita para a esquerda e da esquerda para a direita e a sequência de caracteres é a mesma, como em arara, ovo, subinoonibus, por exemplo.
A classe PalindromoController no package controller deve ter 2 métodos:

- O método invertePalavra que recebe um String e retorna o String invertido. Os métodos push(), pop(), isEmpty() devem ser usados para esse fim;
- O método comparaPalavras, que recebe o String, o String já invertido e retorna um boolean (True para Palíndromo e False para Não Palíndromo)

A classe Principal, no package view, no seu método Main, deve exibir a possibilidade de o usuário inserir uma cadeia de Strings e retornar a ele se é ou não um palíndromo.

4. Implementar um novo projeto Java com a biblioteca PilhaInt.
Esse projeto deve implementar uma solução para uma calculadora em Notação Polonesa Reversa (NPR), também conhecida como posfixa. Calculadoras HP, como a 48G ou a 12C utilizam esse formato de cálculo, em detrimento da maneira algébrica (infixa).
A lógica da NPR se dá como a seguir:

Notação Polonesa Reversa:
(O vídeo https://www.youtube.com/watch?v=-b-f9-9_xAI mostra a HP 50G em operações infixa e posfixa)

- Enquanto for digitado número, ele será empilhado.
- Quando for digitada uma operação (+,-,*,/), 2 valores devem ser desempilhados, se faz a operação com eles e o resultado retorna à pilha
- É importante verificar que a pilha deve ter, no mínimo 2 valores para fazer a operação

O projeto deve ter uma classe de controle (NPRController) que inicializa uma nova Pilha e deve ter duas operações:

- Operação insereValor(Pilha p, int valor):void, faz um push() na pilha

- Operação npr(Pilha p, String op):int. O método deve verificar se a String se trata de uma operação (+,-,*,/), verifica se é possível fazer 2 pop() e, em sendo possível, fazer os 2 pop(), fazer a operação, gravar em uma variável resultado (que é o retorno da operação) e fazer o push() do resultado.
    - Para operações de subtração e divisão (que a ordem importa), fazer o valor do 2º pop() operação valor do 1º pop(), ou seja o valor mais antigo à esquerda da operação
    - Se não houverem 2 valores, deve-se lançar um Exception de pilha com valores insuficientes

A classe view Principal, deve inicializar a pilha e solicitar dados (número ou operação) ao usuário até alguma condição de encerramento, definido por você.

Dica: Para inverter, pode-se usar os métodos substring ou charAt

5. Considerando a biblioteca PilhaString, já criada, faça:
Criar um projeto Java (HistoricoSim) e importe a biblioteca PilhaStrings. Esse novo projeto simulará uma função presente nos navegadores Web, a função de histórico.
A função de histórico empilha os endereços de sites acessados.
A classe Principal, no package view, deve ter na Main, a criação de uma Pilha denominada histórico e deve dar ao usuário a possibilidade de inserir um novo endereço no histórico, remover o último endereço da pilha, saber qual foi o último endereço visitado. Um menu deve ser criado.
A classe HistoricoController deve ter os métodos de validação das operações oferecidas na Main da Classe Principal. Todos os métodos devem receber a pilha criada no método Main como parâmetro.

- O método de inserir um novo endereço, deve-se verificar antes, se o endereço é válido (um endereço válido começa com http:// e, na sequência, deve ter algo no formato www.endereço.com (podendo ser, .com, .co.uk, .com.br, etc.). Não serão aceitos sites sem www;
- O método de remover o último endereço deve dar um erro se o histórico estiver vazio ou desempilhar o último endereço;
- O método de consultar o último endereço, deve dar um erro se o histórico estiver vazio ou apresentar o último endereço, sem removê-lo.

6. Considere a estrutura de pilha abaixo e, apresente uma solução para, usando no máximo uma variável auxiliar, inverter o conteúdo da pilha. O número de pilhas auxiliares possíveis é ilimitado, ou seja, você pode inicializar quantas pilhas auxiliares desejar para resolver o problema, no máximo, uma variável escalar. Note-se que, a pilha original é a pilha que deve apresentar os valores invertidos. Demonstrar com teste de mesa.