PUC Minas

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Curso : Engenharia de Software
Disciplina : Algoritmos e Estruturas de Dados II
Professora : Eveline Alonso Veloso

Exercícios sobre Pilhas

Considere uma pilha de caracteres. Cada item dessa pilha armazena apenas um caractere.

Os exercícios 1 a 4, abaixo, devem ser resolvidos na classe `PilhaCaractere` disponibilizada pela professora.

1) Implemente, em Java, o método `public Boolean verificarExistencia (char l)`, que localiza, na pilha, o item cuja letra corresponde à que foi passada como parâmetro para esse método. Se o item tiver sido localizado na pilha, esse método deve retornar `true`. Caso contrário, esse método deve retornar `false`.

2) Implemente, em Java, o método `public void concatenar (PilhaCaractere pilha)`, capaz de concatenar, à pilha original, a pilha passada como parâmetro.

3) Implemente, em Java, o método `public PilhaCaractere copiar()`, capaz de fazer e retornar uma cópia exata da pilha. Dica: utilize, na implementação desse método, outros métodos já implementados no TAD `PilhaCaractere`.

4) Implemente, em Java, o método `public void imprimir()`, que imprimi, para todos os itens armazenados na pilha, sua letra. A ordem de impressão deve ser do fundo da pilha para o topo. Dica: utilize, na implementação desse método, outros métodos já implementados no TAD `PilhaCaractere`.

Após a implementação dos exercícios 1 a 4, o método `main(String[] args)`, a seguir, deverá funcionar:

```
public class TestaPilha {

   public static void main(String[] args) {

      PilhaCaractere minhaPilha = new PilhaCaractere();
      PilhaCaractere pilha2 = new PilhaCaractere();
      Caractere aux;

      aux = new Caractere('A');
      minhaPilha.empilhar(aux);
      aux = new Caractere('E');
      minhaPilha.empilhar(aux);
      aux = new Caractere('D');
      minhaPilha.empilhar(aux);
      aux = new Caractere('s');
      minhaPilha.empilhar(aux);
      aux = new Caractere('I');
      minhaPilha.empilhar(aux);
      aux = new Caractere('I');
      minhaPilha.empilhar(aux);
```

```
        minhaPilha.desempilhar();
        minhaPilha.desempilhar();

        aux = new Caractere(' ');
        pilha2.empilhar(aux);
        aux = new Caractere('I');
        pilha2.empilhar(aux);
        aux = new Caractere('I');
        pilha2.empilhar(aux);

        minhaPilha.concatenar(pilha2);
        minhaPilha.imprimir(); // AEDs II

        pilha2 = minhaPilha.copiar();
        pilha2.desempilhar();
        pilha2.desempilhar();
        pilha2.desempilhar();
        pilha2.desempilhar();
        pilha2.imprimir(); // AED

        if (minhaPilha.verificarExistencia('A'))
        {
          System.out.println("A letra 'A' foi localizada na pilha."); // essa mensagem deve ser exibida.
        }
        else
        {
          System.out.println("A letra 'A' não foi localizada na pilha");
        }

        if (minhaPilha.verificarExistencia('c'))
        {
          System.out.println("A letra 'c' foi localizada na pilha.");
        }
        else
        {
          System.out.println("A letra 'c' não foi localizada na pilha"); // essa mensagem deve ser exibida.
        }
    }
}
```

5) Considere os TADs Fila e Pilha implementados por meio de estruturas auto-referenciadas. Implemente um método que receba uma fila como parâmetro e retorne uma nova pilha que represente o inverso da fila original, isto é, com todos os seus elementos a serem retirados na ordem inversa a que seriam retirados da fila. O nome desse método será `cloneFilaParaPilhaInvertida`.