

Disciplina: Estruturas de Dados e Algoritmos I-2016/2017

Prova: Teste 1 (20-10-2016)

Esta prova tem a duração de **2 horas** e é **sem consulta**. Identifique TODAS as folhas de teste.

- Apresente uma implementação de **Queue(s)** que use **Stack(s)**. Resolva o exercício no teste completando o código seguinte:

```

public class QueueWithStack<E>{
    /* variáveis de instancia */

    public QueueWithStack( ){
        /* Construtor que cria uma fila de tamanho pré definido*/
    }

    public QueueWithStack(int n ){
        /* Construtor que cria uma fila de tamanho n*/
    }

    public void enqueue(E x){
        /* adiciona x no fim da fila*/
    }

    }

    public E dequeue(){
        /* Remove o 1º elemento da fila e devolve-o*/
    }

    public E front(){
        /* Devolve o 1º elemento da fila*/
    }

}

public boolean isEmpty(){
    /* está vazia?*/
}

}

public int size(){
    /* Retorna o número de elementos da fila */
}

```

- (a) Apresente no código implementado a complexidade de todas as operações, justificando.

2. Numa pequena estação ferroviária a entrada e saída de comboios é realizada usando o esquema representado na figura 1. Suponha que quatro carruagens numeradas de 1 a 4 estão posicionadas à entrada da estação tal como se apresenta na figura. Suponha possíveis

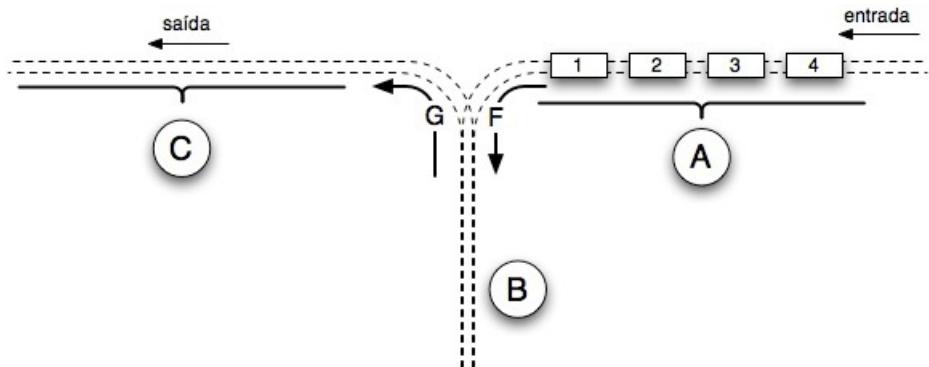


Figure 1: Estação de Comboios

as seguintes acções:

- ADICIONAR(carruagem), que permite a entrada duma carruagem na estação pela zona A,
- RETIRAR(), que permite que uma carruagem saia da estação (pela zona C),
- F() que permite que uma carruagem passe da zona A para a zona B,
- G() que permite que uma carruagem passe da zona B para a zona C.

A sequência de acções: ADICIONAR(1), ADICIONAR(2),ADICIONAR(3), ADICIONAR(4) F(),F(),F(),G(),G(),G(),RETIRAR(),RETIRAR(), RETIRAR(),RETIRAR(), permitirá retirar da estação as carruagens pela ordem 4,3,2,1.

- (a) Apresente um programa em Java que modele o comportamento da estação. Escrevendo somente os códigos ausentes no código seguinte:

```
public class Estacao{
    /* variáveis de instacia */
    _____ zonaA;
    _____ zonaB;
    _____ zonaC;

    public void ADICIONAR(      ){
        }

    public void RETIRAR(){

    }

    public void F(){

    }

    public void G(){

    }
}
```

3. Apresente o código java para o método *maxOf* que retorna o maior elemento da stack passada por argumento. A stack deve ficar inalterada.

```
public static <E extends Comparable<? super E>> E maxOf(Stack<E> s){
```

```
}
```

(a) Qual a complexidade do método? Justifique.

4. Usando o método apresentado nas aulas , avalie a seguinte expressão, dada em postfix. Apresente na resolução, a pilha de execução da conversão.

```
19 2 + 4 3 + / 24 3 3 + /
```