

Aula Prática 9

Resumo:

- Pilhas (*stacks*) e filas (*queues*).

Exercício 9.1

Faça um programa que detecte se uma sequência de letras e dígitos é um palíndromo (ou seja, se a sequência lida do início para o fim é igual à sequência lida do fim para o início). O programa deve ignorar todos os caracteres que não sejam letras ou dígitos, assim como ignorar a diferença entre maiúsculas e minúsculas. Por exemplo as frases: "somos" e "O galo nada no lago" são palíndromos.

Utilize uma estrutura do tipo **Pilha** e outra do tipo **Fila** para resolver este problema.

Exercício 9.2

Escreva um programa que permita visualizar, passo a passo, a resolução do problema das Torres de Hanói. Para esse fim utilize um módulo de **Pilha** (acrescido com um método apropriado de escrita de toda a pilha numa linha) como representação interna de cada torre.

A escrita das torres pode ter o seguinte aspecto:

TA 5 4 3 2 1	TA 3 1	TA
TB	TB 5 4	TB 5 4 3 2 1
TC	TC 2	TC

Para poder visualizar a resolução do problema passo a passo utilize, por exemplo, uma instrução de leitura de uma linha `in.nextLine()` (ignorando essa entrada).

Sugestão: A resolução do problema fica facilitada se criar um módulo **TorresDeHanoi** (inicializado com o número de discos n), com três atributos (internos) do tipo pilha (um para cada torre), e com um serviço de escrita de todas as torres (para além, é claro, do serviço que resolve o problema, cujo algoritmo é idêntico ao feito na aula de recursividade).

Exercício 9.3

Construa uma calculadora com as quatro operações aritméticas básicas que funcione com a notação pós-fixa (*Reverse Polish Notation*¹). Esta notação dispensa a utilização de parêntesis e tem uma implementação muito simples assente na utilização de uma pilha de números reais. Sempre que aparece um operando (número) ele é carregado para a pilha. Sempre que aparece um operador (binário), são retirados os dois últimos números da pilha (se não existirem temos um erro sintático na expressão) e o resultado da operação é colocada na PILHA.

Implemente este programa por forma a que os operandos e os operadores sejam palavras (strings separadas por espaços) lidas do *standard input*. Exemplo de utilização:

```
$ echo "1 2 3 * +" | java -ea p83
Stack: 1.0
Stack: 1.0 2.0
Stack: 1.0 2.0 3.0
Stack: 1.0 6.0
Stack: 7.0
```

¹Nesta notação os operandos são colocados antes do operador. Assim $2 + 3$ passa a ser expresso por $2\ 3\ +$.