Aula Prática 12

Resumo:
- Aplicações de dicionários;
- Dicionários implementados como tabelas de dispersão.

Exercício 12.1
Acrescente à classe `HashTable` do módulo `p2utils` os seguintes métodos:

- `keys()` - devolve um array com todas as chaves existentes na tabela de dispersão.
- `toString()` - Devolve uma representação da tabela de dispersão em cadeia de caracteres de acordo com o seguinte formato: $\{(k_1, e_1), ..., (k_n, e_n)\}$.

Exercício 12.2
O supermercado *DaEsquina* aceita encomendas e faz as respectivas entregas ao domicílio. No entanto, o gerente sente alguma dificuldade no processamento das encomendas bem como na previsão das necessidades de armazém. Decidiu por isso instalar um sistema automático de registo de encomendas que permita, não só registar encomendas recebidas e dar baixa de encomendas entregues, mas também saber em cada momento quantas unidades de cada produto estão pedidas. As encomendas, representadas pela classe `Order` (disponibilizada em anexo), são processadas por ordem de chegada. Implemente assim uma classe `SupermarketOrdering` com os seguintes métodos:

- `enterOrder(order)` - regista uma nova encomenda;
- `serveOrder()` - dá baixa da encomenda mais antiga e devolve-a;
- `query(product)` - devolve o número de unidades de um dado produto que estão pedidas nas encomendas actuais;
- `displayOrderedProducts()` - imprime o número total de unidades encomendadas para cada produto;

Os métodos `enterOrder`, `serveOrder` e `query` devem ter uma complexidade temporal $O(1)$ no número de encomendas. O programa `P122.java` permite testar o módulo pedido.

Exercício 12.3
O ficheiro `numbers.txt` contém uma lista de números com as suas representações numéricas e as suas descrições por extenso. Fazendo uso de um *array* associativo, escreva um programa que traduza, palavra a palavra, todas as ocorrências por extenso de números pelo respectivo valor numérico (mantendo todas as restantes palavras). Exemplo de utilização:

```
$ echo "A list of numbers: eight million two hundred thousand five hundred twenty-four" | java -ea P123
A list of numbers: 8 1000000 2 100 1000 5 100 20 4
```

Exercício 12.4
Utilizando o *array* associativo do exercício anterior, construa um programa que converta um texto representando um número, para o respectivo valor numérico. Por exemplo:

```
$ echo "eight million two hundred thousand five hundred twenty-four" | java -ea P124
eight million two hundred thousand five hundred twenty-four -> 8200524

$ echo "two thousand and thirty three" | java -ea P124
two thousand and thirty three -> 2033
```

Tenha em consideração as seguintes regras na construção do algoritmo:

- Os números são sempre descritos partindo das maiores ordens de grandeza para as mais pequenas (*million, thousand, ...*;
- Sempre que descrições consecutivas de números se fazem por ordem crescente (*eight million*, ou *two hundred thousand*), o valor respectivo vai sendo acumulado por multiplicações sucessivas ($8 * 1000000$, e $2 * 100 * 1000$);
- Caso contrário, o valor acumulado é somado ao total.

Não tenha em consideração o problema de validar a sintaxe correcta na formação de números (por exemplo: *one one million*, *eleven and one*, ...).