

Prática 11

Tópicos

- Ficheiros de texto

Exercício 11.1

Construa um programa que leia um ficheiro de texto e que conte todos os pares de palavras encontrados no ficheiro e o número de ocorrências de cada par. Despreze todas as palavras de tamanho inferior a 3 e considere como separadores os seguintes caracteres:

```
\t\n.,: ' ' ; ? ! - * { } = + & / ( ) [ ] " " ' \ ' "
```

Se utilizar a primeira frase deste problema o resultado no ficheiro de saída deverá ser o seguinte:

```
cada={par=1}  
como={separadores=1}  
considere={como=1}  
construa={programa=1}  
conte={todos=1}  
despreze={todas=1}  
encontrados={ficheiro=1}  
ficheiro={número=1, texto=1}  
inferior={considere=1}  
leia={ficheiro=1}  
número={ocorrências=1}  
ocorrências={cada=1}  
palavras={encontrados=1, tamanho=1}  
par={despreze=1}  
pares={palavras=1}  
programa={que=1}  
...
```

Para cada palavra encontrada no ficheiro o programa deverá associar um conjunto de todas as palavras seguintes contando igualmente quantas vezes cada par ocorre.

Teste o programa com o ficheiro “major.txt”. O resultado deverá ser semelhante a:

```
1864={tratava=1}  
abacateiros={entoando=1, mangueiras=1, oitenta=1, suas=1, troncos=1}  
abacates={ora=1, pouco=1}  
...  
voltava={aos=1, biblioteca=1, com=1, ficava=1, idéia=1, olhava=1, para=1, seu=1}  
voltavam={mesmo=1, oficial=1, para=1, sorridentes=1}  
voltou={acidente=1, agarrou=1, alegria=1, aos=1, bonde=1, general=1, instante=1, lhe=1}
```

Exercício 11.2

Crie um programa para gerir as notas dos estudantes de uma turma. O programa deve ler as informações dos alunos de um ficheiro, que contém o nome do aluno e as suas notas.

Usando a Java Collection que achar mais adequada o programa deve gerir a informação de todos os alunos.

Como ficheiro com a informação dos alunos pode usar o ficheiro “alunos.txt” que providencia a avaliação de uma turma de alunos.

A classe *Student*, a interface *GradeCalculator* e a classe *GradebookTester* já são fornecidas.

Deve criar as seguintes classes de forma que *GradebookTester* funcione.

- *Gradebook*: esta classe faz a gestão da informação de todos os alunos. A classe deve conter:
 - Método *load* que recebe o nome de um ficheiro e lê todos os dados contidos no ficheiro de texto.
 - Métodos *removeStudent* e *addStudent*, que remove/adiciona um estudante à classe.
 - Método *getStudent*, que devolve um objeto estudante que contém as informações do estudante que foi pedido.
 - Método *calculateAverageGrade*, calcula a média de um estudante.
 - Método *PrintAllStudents*, que imprime as informações de todos os estudantes.
- *SimpleGradeCalculator*: Implementa a interface *GradeCalculator* e calcula a média das notas.

Se achar necessário pode fazer alterações à classe *Student*.

O ficheiro “alunos.txt” contém um aluno por linha com a seguinte informação:

Nome do aluno | nota 1 | nota 2 | nota 3

Exercício 11.3

Uma companhia energética quer automatizar o processo de geração dos relatórios de consumo dos clientes. Para isso é necessário ler um ficheiro com os dados de leitura dos vários clientes e gerar o relatório final com o consumo total de cada cliente.

Sendo assim, é preciso criar um programa que leia o ficheiro “clientes.txt”; as classes *Customer* e *EnergyUsageReportTester* já são fornecidas. Deve criar a classe *EnergyUsageReport*, que fornece os seguintes métodos:

- *load*: lê o ficheiro com os dados dos clientes e guarda os valores numa Java Collection;
- *addCustomer / removeCustomer*: adiciona/remove um cliente;
- *getCustomer*: providencia os dados de um determinado cliente;
- *calculateTotalUsage*: Calcula o uso total de energia de um cliente;
- *generateReport*: Gera o relatório onde é listado em cada linha o ID do cliente e a energia consumida.

O ficheiro “clients.txt” contém um cliente por linha com a seguinte informação:

ID cliente | leitura 1 | leitura 2 | leitura 3 | leitura 4

Exercício 11.4 (Extra)

No dossier da disciplina encontra, para além deste guião, dois ficheiros de texto: *voos.txt* e *companhias.txt*.

O primeiro representa os voos que chegaram ao aeroporto do Porto no dia 24 de Maio. A estrutura deste ficheiro é a seguinte (os campos são separados por *tab*):

Hora	Voo	Origem	Atraso
00:50	TP 1944	Lisboa	
07:00	AEA1147	Madrid	
07:35	IB 8720	Madrid	00:25
...			

O segundo contém uma tabela com as siglas e os nomes de cada companhia:

Sigla	Companhia
A5	HOP!
AE	Air Europa
DT	TAAG
...	

Construa um programa que leia estes dois ficheiros para estruturas adequadas. Crie a classe *Voo*, por exemplo, use conjuntos para armazenar os voos em memória, bem como outras estruturas/algoritmos que ache necessários para cada uma das alíneas seguintes.

- a) Apresente no ecrã a lista de voos com informação mais completa, tal como consta da tabela seguinte:

Hora	Voo	Companhia	Origem	Atraso	Obs
00:50	TP 1944	TAP Portugal	Lisboa		
07:00	AEA1147	Air Europa	Madrid		
07:35	IB 8720	Iberia	Madrid	00:25	Previsto: 8:00
07:35	TO 3408	Transavia France	Paris, Orly		
07:40	FR 5451	Ryanair	Faro		
07:55	EZY3771	EasyJet Airlines	Paris, Ch. de Gaulle	00:33	Previsto: 8:28
...					

- b) Guarde a tabela no ficheiro *Infopublico.txt*.
c) Calcule a média dos atrasos por companhia e apresente no ecrã uma tabela (Companhia, Atraso médio) ordenada por ordem crescente de atraso médio.
d) Guarde no ficheiro *idades.txt* uma tabela com informação com o total de chegadas de cada cidade origem. Exemplo (ordenação por número de voos):

Origem	Voos
Lisboa	11
Madrid	9
Paris, Orly	8
...	