universidade de aveiro deti departamento de electrónica, telecomunicações e informática

Programação I

Folha Exercícios 12

António J. R. Neves
João Rodrigues
Osvaldo Pacheco
Arnaldo Martins

2018/19

Folha exercícios 12

O objetivo desta folha é fornecer mais exercícios para auxiliar a preparação para o **exame prático final**. Além destes, sugere-se a resolução de exercícios dos guiões das aulas práticas, particularmente os das últimas aulas.

Problema 1

Pretende-se implementar um programa que permita processar informação de uma estação meteorológica ao longo de um mês, nomeadamente os valores da temperatura e humidade relativa do ar às 12h de cada dia. No máximo, o programa deve ser capaz de processar 31 medidas.

O uso de estruturas de dados (classes) e funções será valorizado na resolução deste problema.

O programa deve funcionar de forma repetitiva com base no menu de opções que a seguir se apresenta:

```
Estação meteorológica:
  1 - Ler ficheiro de dados
  2 - Acrescentar medida
  3 - Listar valores de temperatura e humidade
  4 - Listar medidas ordenadas por valor de temperatura
  5 - Listar medidas ordenadas por valor de humidade
  6 - Calcular valores médios de temperatura e humidade
  7 - Calcular valores máximos e mínimos de temperatura e
humidade
  8 - Calcular histograma das temperaturas e humidade
  9 - Terminar o programa
  Opção ->
```

O programa deverá permitir as seguintes operações:

1) Ler um ficheiro de texto contendo os valores da temperatura e humidade. O nome do ficheiro deve ser pedido ao utilizador. Considere que o ficheiro contém o valor da temperatura seguido do valor da humidade, ambos valores inteiros. Considere, também, que o ficheiro contém apenas valores válidos e o número de medidas é inferior ou igual a 31.
2) Introdução da informação associada a uma nova medida. O programa deve aceitar apenas temperaturas no intervalo [-10, 40] e valores de humidade relativa no intervalo [0, 100]. Deve ter em atenção o número máximo de medidas permitidas.
3) Mostrar ao utilizador a informação sobre todas as medidas.
4) Mostrar ao utilizador a informação sobre todas as medidas, ordenadas por ordem crescente de temperatura.

5) Mostrar ao utilizador a informação sobre todas as medidas, ordenadas por ordem decrescente de humidade.
6) Calcular e mostrar ao utilizador os valores médios da temperatura e humidade.
7) Calcular e mostrar ao utilizador os valores máximos e mínimos da temperatura e humidade.
8) Mostrar ao utilizador um histograma horizontal (contagem do número de ocorrências de cada valor) para a temperatura e outro para a humidade. Os histogramas devem ter o seguinte aspeto:

```
Histograma da temperatura
------------------------
-10 | ****
 - 9 |
...
  0 | **
...
 39 | *
 40 | **


Histograma da humidade
----------------------
  0 | ****
  1 |
...
 99 | *
100 | **
```

Problema 2

Pretende-se implementar um programa que permita gerir uma prova automóvel com no máximo 10 participantes. Para cada participante (classe Piloto) deverá ser guardada a seguinte informação:

número da viatura: valor inteiro incrementado automaticamente após cada introdução;

nome do piloto: texto livre;

matrícula da viatura: texto com 8 carateres representando uma matrícula portuguesa;

O programa deve funcionar de forma repetitiva com base no menu de opções que a seguir se apresenta:

```
Gestão de uma prova automóvel:
1 - Inserir informação dos pilotos
2 - Listar pilotos ordenados pelo número da viatura
3 - Alterar informação de um piloto
4 - Remover piloto com base no número da viatura
5 - Listar pilotos ordenados pelo nome
6 - Remover piloto(s) com base no nome
```

```
7 - Validar matriculas
8 - Terminar o programa
Opção ->
```

O programa deverá permitir as seguintes operações:

1) Introdução da informação associada aos pilotos, terminando com a introdução de um nome vazio, isto é, nome com zero carateres. Toda a informação deverá ser pedida ao utilizador com a exceção do número do veículo que deverá ser preenchido pelo programa, assumindo um valor incremental começando em 1 para o primeiro piloto. Não é necessário fazer validações na introdução dos dados. Nesta opção, a base de dados deve ser preenchida desde o início, ignorando os dados previamente introduzidos.
2) Mostrar ao utilizador a informação sobre os pilotos, ordenada por ordem decrescente do número da viatura.
3) Alterar a informação sobre um piloto, dado o número da viatura pedido ao utilizador. Caso a viatura exista, deverá ser pedido ao utilizador todos os campos do piloto com a exceção do número da viatura. Deve informar o utilizador se a viatura não existir.
4) Remover um piloto, dado o número da viatura pedido ao utilizador. Deve informar o utilizador se a viatura não existir.
5) Mostrar ao utilizador a informação sobre os pilotos, ordenada lexicograficamente por ordem crescente do seu nome.
6) Remover um ou vários pilotos com base no nome, parcial ou completo, pedido ao utilizador. Sugere-se a utilização da função **indexOf** da classe String. No caso de haver mais do que um piloto que respeite a condição, deve ser perguntado ao utilizador qual ou quais os pilotos a remover.
7) Verificação das matrículas dos veículos presentes na prova e, em caso de matrícula inválida, pedir ao utilizador nova matrícula para o piloto em causa. Considere as três possibilidades de matrículas em Portugal: AA-00-00, 00-00-AA e 00-AA-00.
8) Terminar o programa.

Problema 3

O Concurso Micro-Rato é uma competição robótica que se realiza regularmente desde 1995, organizada pelo Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro. É uma competição entre pequenos robôs móveis e autónomos que devem cumprir dois objetivos em sequência: o primeiro objetivo é ir desde a "área de partida" até à "área de farol"; o segundo objetivo consiste em regressar à área de partida.

Neste exame deve desenvolver um programa para a gestão de uma prova do concurso Micro-Rato. Para cada robô em prova deverá ser guardada a seguinte informação:

Nome do robô - texto livre;
Tempo de prova em segundos - valor inteiro;

Número de elementos da equipa - valor inteiro

O programa deve funcionar de forma repetitiva com base num menu de opções que a seguir se apresenta:

```
Micro-Rato 2013 - Gestão da prova:
1 - Adicionar informação relativa a um robô
2 - Imprimir informação dos robôs em prova
3 - Vencedor da prova e tempos médios de prova
4 - Média do número de elementos por equipa
5 - Mostrar o nome dos robôs em maiúsculas
6 - Alterar informação de um robô
7 - Remover robôs da competição
8 - Gravar informação da prova num ficheiro
9 - Terminar o programa
Opção ->
```

O programa deverá permitir as seguintes operações:

1) Introdução da informação associada a um novo robô. Sempre que o utilizador escolher esta opção, deverá ser adicionado um novo robô à prova.
2) Mostrar informação sobre os robôs em prova. Nesta alínea o tempo de prova deve ser apresentado no formato HH:MM:SS.
3) Mostrar ao utilizador a informação sobre o robô vencedor que corresponde ao robô que demorou menos tempo a realizar a prova. Deverá também calcular e imprimir o tempo médio obtido pelos robôs na prova.
4) Calcular e imprimir a média do número de elementos por equipa.
5) Mostrar ao utilizador o nome dos robôs em prova, que deverá aparecer todo em maiúsculas. Os nomes devem aparecer ordenados lexicograficamente por ordem crescente.
6) Alterar a informação sobre um robô, dado o seu nome, que será pedido ao utilizador. Caso o robô exista, deverão ser pedidos ao utilizador todos os campos do mesmo. Deve informar o utilizador se o robô não existir.
7) Remover os robôs em prova cujo tempo gasto seja superior a um determinado valor, pedido ao utilizador.
8) Gravar num ficheiro, cujo nome seja pedido ao utilizador, a informação sobre os robôs em prova.
9) Terminar o programa.

Problema 4

Pretende-se neste problema desenvolver um programa que permita gerir a pauta de notas de uma disciplina com vista à sua submissão nos Serviços Académicos. A informação relativa ao número mecanográfico dos alunos de uma determinada disciplina já se encontra num ficheiro previamente fornecido, sendo necessário pedir ao utilizador a informação sobre a nota da época normal e a nota da época de recurso de cada aluno. A nota final deve ser calculada pelo programa, considerando a melhor das duas. Tenha em atenção que um aluno pode faltar à época normal, à época de recurso ou a ambas.

O programa deve funcionar de forma repetitiva com base no menu de opções que a seguir se apresenta:

```
Serviços Académicos - Gestão de uma pauta:
    1 - Ler ficheiro com números mec. e pedir informação de
notas
    2 - Imprimir no terminal a informação da disciplina
    3 - Calcular estatísticas das notas finais
    4 - Alterar as notas de um aluno
    5 - Imprimir um histograma de notas
    6 - Gravar num ficheiro a informação da disciplina
(ordenada)
    7- Terminar o programa
    Opção ->
```

O programa deverá permitir as seguintes operações:

1) Pedir ao utilizador o nome do ficheiro com a informação relativa ao número mecanográfico dos alunos da disciplina. Considere que os números existentes no ficheiro são todos válidos.
   Para cada aluno, deve pedir ao utilizador as notas obtidas nas épocas normal e recurso. Tenha em atenção que só devem ser consideradas válidas as notas com valor inteiro no intervalo [0...20] e o valor 77 como código para faltou.
2) Mostrar ao utilizador a informação sobre todos os alunos da disciplina. Deve mostrar a informação sob a forma de tabela como apresentado de seguida.

```
Opção -> 2
Pauta de uma disciplina
-------------------------
|  18000 | 10 | 12 | 12 |
|   1600 | 77 | 12 | 12 |
|  13000 | 12 | 77 | 12 |
|    500 | 77 | 77 | 77 |
|     30 |  9 |  3 |  9 |
|  70000 | 18 | 20 | 20 |
-------------------------
```

3) Calcular e imprimir no terminal o valor da nota final média de todos os alunos, a informação do melhor aluno, o número de alunos aprovados e reprovados.
4) Alterar a informação sobre um aluno, dado o seu número mecanográfico pedido ao utilizador. Caso o aluno exista, deverá ser pedido ao utilizador o valor das suas notas das épocas normal e de recurso. Deve informar o utilizador se o aluno não existir.
5) Mostrar ao utilizador um histograma vertical referente às notas finais, como apresentado no exemplo de utilização em anexo.

Opção -> 5

```
                    Histograma de uma disciplina

                                        *
                                        *
                       *                *                        *
------------------------------------------------------------------------
 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

6) 6. Gravar num ficheiro a informação sobre todos os alunos **aprovados** à disciplina, ordenados por ordem decrescente da sua nota final.
7) 7. Terminar o programa.

Problema 5

Considere o ficheiro temperaturas.txt com o formato indicado. Este ficheiro tem uma lista de temperaturas de várias cidades com a seguinte estrutura: na primeira linha tem um valor inteiro com o número de temperaturas a ler; cada uma das restantes linhas começam com um número inteiro positivo que indica o dia, um segundo valor que indica a temperatura em graus C (entre -50 e 50), e o resto da linha o nome da cidade, de acordo com o exemplo:

```
5
1 33 Porto
56 26 Vila Real
1303 33 Aveiro
24 -13 Moscovo
119 35 Los Angeles
```

Cada cidade é representada pela classe **Cidade {int dia; int temperatura; String nome}**.
Complete o programa dado implementando cada uma das funções indicadas:

1) Função para ler os dados do ficheiro para um array do tipo Cidade, devolvendo o array. Tem como argumento o nome do ficheiro.

   static Cidade[] lerFichTemp(String nomeF) throws IOException{}

2) Função, que tendo um array de cidades e uma frase como argumentos imprime todas as cidades que contenham a frase (procura no dia, temperatura e nome – crie um string com os 3 campos).

static void procurarListar(Cidade[] c, String frase){}

Exemplo da procura de "**13**":	Exemplo da procura de "**os**":
1303 11 Aveiro 24 -**13** Moscovo	24 -13 M**os**covo 119 35 L**os** Angeles

3) Função para calcular a temperatura máxima e mínima, devolvendo um array de 2 elementos, em que o 0 tem o índice da cidade com a temperatura máxima e o 1 tem o índice da cidade com a temperatura mínima. O argumento é o array com as cidades.
   static int[] maxMin(Cidade[] c){}

4) Função para gravar num ficheiro de texto os registos de valores de uma determinada cidade. Os argumentos são o array de cidades e o nome da cidade a gravar (ex: se o nome for "Aveiro" cria o ficheiro "Aveiro.txt");
   static void gravarCidade(Cidade[] c, String nome) throws IOException{}

5) Função para calcular e devolver um array com a frequência de temperaturas (número de vezes que uma determinada temperatura aparece). Deve ignorar temperaturas fora do intervalo [-50,50]. O argumento á o array de cidades.
   static int[] freqTemp(Cidade[] c){}
6) Função para imprimir o array da frequência de temperaturas, dado como argumento. Só imprime os valores > 0.
   static void printFreq(int[] f){}

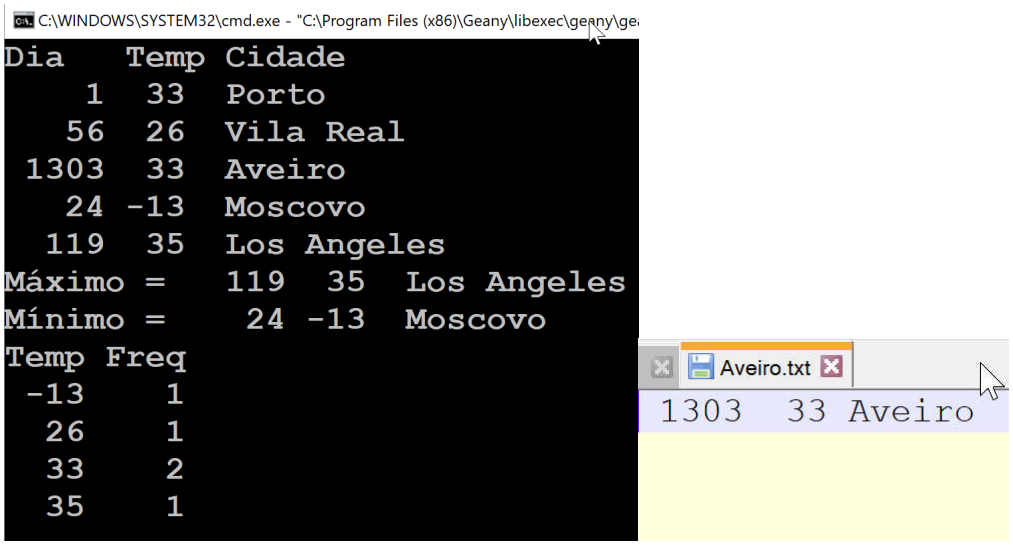
programa para completar:

```
import java.util.Scanner;
import java.io.*;

public class Teste2A {

    public static void main(String[] args) throws IOException{
        Cidade[] cidades;
        cidades = lerFichTemp("temperaturas.txt");
        procurarListar(cidades," "); // espaço lista tudo
        int[] maxmin = maxMin(cidades);
        System.out.printf("Máximo = %5d %3d %s%n", cidades[maxmin[0]].dia,
cidades[maxmin[0]].temperatura, cidades[maxmin[0]].nome);
        System.out.printf("Mínimo = %5d %3d %s%n", cidades[maxmin[1]].dia,
cidades[maxmin[1]].temperatura, cidades[maxmin[1]].nome);
        gravarCidade(cidades, "Aveiro");
        int[] freq = freqTemp(cidades);
        printFreq(freq);
    }
// Implementar funções pedidas aqui
}
class Cidade {
    int dia;
    int temperatura;
    String nome;}
```

Resultados:


C:\WINDOWS\SYSTEM32\cmd.exe - "C:\Program Files (x86)\Geany\libexec\geany\ge
Dia Temp Cidade
1 33 Porto
56 26 Vila Real
1303 33 Aveiro
24 -13 Moscovo
119 35 Los Angeles
Máximo = 119 35 Los Angeles
Mínimo = 24 -13 Moscovo
Temp Freq
-13 1
26 1
33 2
35 1
Aveiro.txt
1303 33 Aveiro