```java
/*
 * JAM, 19-jan-2018
 * Exame final
 */
import java.util.Scanner;
import java.io.*;

public class ExameFinalB {

    public static void main(String[] args) throws IOException{
        Pais[] paises;
        paises = lerFichHum("PaisesHumid.txt");
        procurarListar(paises," ");  // espaço lista tudo
        procurarListar(paises,"56"); // lista paises onde apareca "56"
        int [] mins = humMin(paises);
        for(int i=0; i<mins.length; i++)
            System.out.printf("Minimo = %5d %3d %s%n", paises[mins[i]].dia,
                              paises[mins[i]].humidade, paises[mins[i]].nome);

        int[] freq = freqHum(paises);
        printFreq(freq);
        // exame final
        // verifica se dia e pais existe
        System.out.println(diaPaisExiste(paises,paises.length,24,"Marrocos"));
        // calcula array sem dia e paises repetidos
        paises=removerRepetidos(paises);

        // ordena por ordem crescente do dia
        bubbleSort(paises);
        // guarda num ficheiro
        gravarPais(paises, "PaisesHumidOrd.txt");
    }
    // Pergunta 1) - função para ler as humidades do ficheiro
    static Pais[] lerFichHum(String nomeF) throws IOException{
        File f = new File(nomeF);
        Scanner lerF = new Scanner(f);
        int nPaises = lerF.nextInt();  // 1ª linha, nº de humidades
        Pais[] p = new Pais[nPaises];
        int n=0;
        while (lerF.hasNextLine() && n < nPaises) {
            p[n] = new Pais();
            p[n].dia = lerF.nextInt();
            p[n].humidade = lerF.nextInt();
            p[n].nome = lerF.nextLine().trim(); // trim() tira espaços no início e fim
            n++;
        }
        lerF.close();
        return p;
    }
    // Pergunta 2) função para procurar uma frase e listar
    static void procurarListar(Pais[] p,String frase) {
        System.out.printf("Dia   Hum Pais%n");
        for (int i=0; i<p.length; i++) {
            String linha = p[i].dia + " " +p[i].humidade+" "+p[i].nome;
            if (linha.indexOf(frase) >=0) {
                System.out.printf("%5d %3d %s%n", p[i].dia, p[i].humidade,
                p[i].nome);
            }
        }
    }
    // Pergunta 3) função para calcular as humidades minimas.
    // devolve um array com os indices dos dias e paises onde ocorreu a humidade minima.
    static int [] humMin(Pais [] p) {
```

```
63          int  [] mm;
64          int min = 0;
65          int cont = 1;
66          for (int i=1; i<p.length; i++)
67              if(p[i].humidade < p[min].humidade) {
68                  min = i;
69                  cont = 1;
70              }
71              else if(p[i].humidade == p[min].humidade)
72                  cont++;
73          mm = new int[cont];
74          int n=0;
75          for(int i=0; i<p.length; i++)
76              if(p[i].humidade == p[min].humidade)
77                  mm[n++] = i;
78          return mm;
79      }
80
81      // Pergunta 4) função para gravar humidades
82      static void gravarPais(Pais[] p, String nome) throws IOException{
83          File f = new File(nome);
84          PrintWriter pf = new PrintWriter(f);
85          pf.println(p.length);
86          for (int i=0; i<p.length; i++) {
87              pf.printf("%5d %3d %s%n", p[i].dia, p[i].humidade, p[i].nome);
88          }
89          pf.close();
90      }
91      // Pergunta 5) função para calcular frequência das humidades
92      // humidade entre 10 e 100 - array de freq entre 0 (10) e 90 (100)
93      static int[] freqHum(Pais[] p) {
94          int[] f = new int[91];
95          for (int i=0; i<p.length; i++) {
96              if (p[i].humidade >= 10 && p[i].humidade <= 100)
97                  f[p[i].humidade-10]++;
98          }
99          return f;
100     }
101     // Pergunta 6) função para imprimir as frequências  > 0
102     static void printFreq(int[] f) {
103         System.out.printf("Hum. Freq%n");
104         for (int i=0; i<f.length; i++) {
105             if (f[i]>0)System.out.printf("%4d %4d%n", i+10, f[i]);
106         }
107     }
108     // exame final
109     // Pergunta 7) função para verificar se um dia e pais existe. false não
        existe; true existe
110     static boolean diaPaisExiste(Pais[] p,int comp, int dia,String pais) {
111         boolean indice = false;
112         for (int i=0; i<comp; i++) {
113             if (p[i].dia == dia && p[i].nome.indexOf(pais) >= 0) {
114                 indice = true;
115             }
116         }
117         return indice;
118     }
119     // Pergunta 8) função para criar array de paises sem dia e pais repetidos
120     //             devolve array com o numero de elementos não repetidos
121     static Pais[] removerRepetidos(Pais [] p) {
122         Pais[] r = new Pais[p.length];
123         int n=0;
124         for (int i=0; i<p.length; i++) {
125             if(!diaPaisExiste(r,n,p[i].dia,p[i].nome)) {
126                 r[n]=p[i];
```

```
127                     n++;
128                 }
129             }
130             // devolve array com o comprimento exato
131             Pais[] u = new Pais[n];
132             System.arraycopy(r,0,u,0,n);
133             return u;
134         }
135         // Pergunta 9) ordena por ordem crescente do dia
136         static void bubbleSort(Pais a[]) {
137             boolean troca;
138             int n=0;    //nº de valores ordenados
139             do {
140                 troca=false;
141                 for(int i = 0 ; i <= a.length - 2 -n; i++) {
142                     if(a[i].dia > a[i+1].dia) {
143                         swap(a, i, i+1);
144                         troca=true;
145                     }
146                 }
147                 n++;
148             } while(troca);
149         }
150 
151         static void swap(Pais a[], int i, int j) {
152             Pais tmp = new Pais();
153             tmp = a[i];
154             a[i] = a[j];
155             a[j] = tmp;
156         }
157 
158     }
159     class Pais {
160         int dia;
161         int humidade;
162         String nome;
163     }
164 
```