Rilevazione di dati in una stazione meteorologica

In una stazione meteorologica vengono misurati i valori della temperatura esterna (-20, 55°C), della pressione atmosferica (750, 1050 millibar) e dell'umidità relativa (0.01, 1.00).
Si desidera conservare il valore delle tre grandezze rilevate ogni ora per ogni giornata di un intero anno solare.

In qualsiasi momento deve essere possibile per un utente ottenere:

- la sequenza dei valori minimi e massimi delle tre grandezze giorno per giorno, sia in forma tabellare che in forma grafica per un periodo compreso fra due date specificate dall'utente stesso;
- la sequenza dei valori delle tre grandezze. sia in forma tabellare che in forma grafica, in una particolare data specificata dall'utente stesso.

Il candidato, facendo riferimento a sistemi di elaborazione di sua conoscenza e specificando le ipotesi aggiuntive che ritenga necessarie, deve:

- definire a grandi blocchi una soluzione globale del problema, sia per quanto riguarda le risorse fisiche, sia per quanto riguarda la procedura;
- scrivere in un opportuno linguaggio di programmazione di sua conoscenza la procedura o una parte di essa a sua scelta.

(Seconda prova scritta di informatica, maturità tecnica industriale, 1983)

TRACCIA PER LA SOLUZIONE

Il problema richiede di conservare i valori rilevati da una stazione meteorologica in tutte le ore di ogni giorno dell'anno. Vista la loro quantità e l'arco di te mpo in cui devono essere conservati, è necessario memorizzare tutti i valori in un file su disco. Il problema può essere risolto scrivendo due programmi separati: uno per l'inserimento dei dati e l'altro per eseguire le interrogazioni.

Il programma di inserimento gestisce il file per le operazioni di scrittura, mentre il programma di interrogazione accede al file in lettura. Dovendo condividere lo stesso file, è necessario che entrambi i programmi interpretino i dati memorizzati sul file utilizzando la stessa struttura.

La struttura del file, che verrà poi ripresa nella descrizione degli oggetti, è composta dai seguenti campi:

giorno (valori da 1 a 31)
mese (valori da 1 a 12)
ora (valori da 0 a 23)
temperatura (valori da -20 a 55)
pressione (valori da 750 a 1050)
umidità (valori da 0.01 a 1.00)

L'insieme di questi campi forma una situazione della stazione meteorologica.

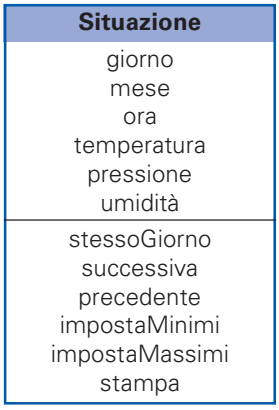
Si suppone che le situazioni vengano memorizzate in ordine di data (giorno e mese), cioè l'utente deve fornire i dati al programma di inserimento in modo ordinato. Questa ipotesi facilita il compito del programma di interrogazione, che esegue solo una ricerca sequenziale e non deve ordinare i dati che trova.

Analizziamo ora quantitativamente le risorse fisiche richieste da questi programmi. Poiché si devono memorizzare su disco i campi precedentemente elencati, l'occupazione di memoria prevista può essere calcolata associando ad ogni campo un tipo di dato di Java e verificando quanti byte sono necessari.

Campo	Tipo di dato	Occupazione
giorno	byte	1 byte
mese	byte	1 byte
ora	byte	1 byte
temperatura	float	4 bytes
pressione	short	2 bytes
umidità	float	4 bytes

Ogni situazione occupa quindi complessivamente 13 bytes. Naturalmente questa è una stima per difetto, perché Java memorizza, oltre al valore dei campi, anche altre informazioni sulla struttura dei dati. Ogni giorno vengono memorizzate 24 situazioni: questo significa che in un anno vengono memorizzate 24x365=8760 situazioni. Moltiplicando il numero di situazioni per la singola occupazione, si ottiene l'occupazione complessiva che è di 8760x13=113880 bytes. Considerando anche le informazioni aggiuntive salvate da Java, si può stimare che verranno occupati qualche centinaia di Kb.
Per l'interfaccia grafica di questo problema vengono utilizzate le componenti del package Swing.

Una classe importante, che viene usata dai programmi di inserimento e interrogazione, è quella che descrive la struttura del file. Saranno gli oggetti derivati da questa classe ad essere memorizzati nel file. Il diagramma della classe *Situazione* è mostrato nella seguente figura:



I sei attributi sono tutti pubblici.
I metodi *stessoGiorno*, successiva e precedente eseguono dei confronti tra la data in cui è stata rilevata questa situazione e una data passata come parametro.
Il metodo *impostaMinimi* e *impostaMassimi* hanno come parametro un'altra situazione. Il loro compito è confrontare le due situazioni e memorizzare in questa i valori minimi oppure i massimi.

Nel **programma di inserimento** vengono richiesti i valori con cui impostare gli attributi di un oggetto di classe *Situazione* che successivamente viene salvato sul file. Poiché viene utilizzata un'interfaccia grafica, si continuano a salvare le situazioni finché non viene chiusa la finestra. La classe viene definita implementando l'interfaccia *Serializable*, per rendere gli oggetti di questa classe serializzabili e quindi memorizzabili su di un file.

IMPLEMENTAZIONE DELLA CLASSE (*Situazione.java*)

```
import java.io.*;

class Situazione implements Serializable
{
   public byte giorno, mese, ora;
   public float temperatura, umidita;
   public short pressione;

   // costruttore vuoto
   public Situazione() {}

   // costruttore con un parametro
   public Situazione(Situazione sit)
   {
      giorno = sit.giorno;
      mese = sit.mese;
      ora = sit.ora;
      temperatura = sit.temperatura;
      umidita = sit.umidita;
      pressione = sit.pressione;
   }

   // controlla se questa situazione e' stata rilevata il gg/mm
   public boolean stessoGiorno(byte gg, byte mm)
   {
      return ((giorno == gg) && (mese == mm));
   }

   // confronta la data di questa situazione con i parametri gg/mm
   public boolean successiva(byte gg, byte mm)
   {
      if (mese > mm)
      {
         return true;
      }
      if ((mese == mm) && (giorno > gg))
      {
         return true;
      }
      // se non si verificano le situazioni precedenti
      return false;
   }

   public boolean precedente(byte gg, byte mm)
   {
      return !successiva(gg,mm);
   }
```

```
    // imposta i valori di T,U,P al minimo tra quello attuale e quello di sit
    public void impostaMinimi(Situazione sit)
    {
      if (sit.temperatura < temperatura)
      {
        temperatura = sit.temperatura;
      }
      if (sit.pressione < pressione)
      {
        pressione = sit.pressione;
      }
      if (sit.umidita < umidita)
      {
        umidita = sit.umidita;
      }
    }

    // imposta i valori di T,U,P al massimo
    // tra quello attuale e quello di sit
    public void impostaMassimi(Situazione sit)
    {
      if (sit.temperatura > temperatura)
      {
        temperatura = sit.temperatura;
      }
      if (sit.pressione > pressione)
      {
        pressione = sit.pressione;
      }
      if (sit.umidita > umidita)
      {
        umidita = sit.umidita;
      }
    }

    // visualizza gli attributi
    public void stampa()
    {
      System.out.println("Temperatura: "+temperatura);
      System.out.println("Pressione: "+pressione);
      System.out.println("Umidita: "+umidita);
    }
  }
```

Di seguito viene mostrata la traccia del programma di inserimento.

PROGRAMMA JAVA (*Inserimento.java*)

```
  import javax.swing.*;

  class Inserimento
```

```
{
   public static void main(String argv[])
   {
      InserFrame f = new InserFrame();
      f.setTitle("Inserimento dati");
      f.pack();
      f.setVisible(true);
   }
}
```

IMPLEMENTAZIONE DELLA CLASSE (*InserFrame.java*)

```
import java.io.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class InserFrame extends JFrame implements ActionListener
{
   private JPanel q = new JPanel();
   private JTextField g = new JTextField(5);
   . . . . .
   . . . . .
   . . . . .

   private JButton OK = new JButton("OK");
   private FileOutputStream f;
   private ObjectOutputStream fOUT;
   public InserFrame() { . . . . }
   public void actionPerformed(ActionEvent e) { . . . . }
}
```

Gli attributi di questa classe sono sia le componenti della finestra che gli oggetti utilizzati per accedere al file.
Il costruttore della finestra è implementato nel seguente modo:

```
   public InserFrame()
   {
      // dispone le componenti nella finestra
      . . .
      // apre il file per l'aggiornamento
      try
      {
         f = new FileOutputStream("situazione.dat", true);
         fOUT = new ObjectOutputStream(f);
      }
      catch(IOException e)
      {
         System.out.println("Eccezione: "+e.getMessage());
         System.exit(1);
      }
   }
```

Il gestore del pulsante *OK* è implementato nel seguente modo:

```
public void actionPerformed(ActionEvent e)
{
  String bottone = e.getActionCommand();
  Situazione sit = new Situazione();

  if (bottone.equals("OK"))
  {
    //legge i valori
    try
    {
      sit.giorno = Integer.valueOf(g.getText()).byteValue();
      sit.mese = Integer.valueOf(m.getText()).byteValue();
      sit.ora = Integer.valueOf(o.getText()).byteValue();
      sit.pressione = Integer.valueOf(p.getText()).shortValue();
      sit.temperatura = Float.valueOf(t.getText()).floatValue();
      sit.umidita = Float.valueOf(u.getText()).floatValue();
    }
    catch(Exception exc)
    {
      return;     // non memorizza la situazione
    }

    //scrive sul file
    try
    {
      fOUT.writeObject(sit);
      fOUT.flush();
    }
    catch(Exception exc)
    {
      System.out.println("Eccezione: "+exc.getMessage());
      return;
    }
  }
}
```

Il gestore del pulsante crea un oggetto di classe *Situazione* e imposta i suoi attributi con i valori inseriti dall'utente. Se i valori inseriti sono corretti, l'oggetto viene memorizzato sul file.

Il **programma di interrogazione** permette di eseguire due interrogazioni diverse con le situazioni memorizzate nel file:

- *interr1*, ricerca le situazioni comprese tra due date, scegliendo per ogni giorno i valori minimi e massimi;
- *interr2*, ricerca le situazioni di una sola giornata.

Le interrogazioni possono essere gestite costruendo una classe opportuna in cui vengono definiti i due metodi *interr1* e *interr2*. Gli attributi sono tre vettori in cui viene inserito il risultato delle interrogazioni.

Interrogazioni
minimi massimi giornata
interr1 interr2

IMPLEMENTAZIONE DELLA CLASSE (*Interrogazioni.java*)

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

class Interrogazioni
{
   public static Vector minimi;
   public static Vector massimi;
   public static Vector giornata;

   // ricerca situazioni comprese tra due date
   public static void interr1(byte g1, byte m1, byte g2, byte m2) { . . . }

   // ricerca situazioni di una giornata
   public static void interr2(byte g, byte m) { . . . }
}
```

Sia gli attributi che i metodi sono stati dichiarati come *static* e possono quindi essere riferiti direttamente senza creare istanze della classe *Interrogazioni*.
Per esempio, per invocare *l'interr2* si usa la seguente istruzione:

```
byte g=2;
byte m=2;
Interrogazioni.interr2(g, m);
```

I metodi *interr1* e *inter2* utilizzano la classe *Situazione* definita in precedenza.

- **Metodo interr1**

```
public static void interr1(byte g1, byte m1, byte g2, byte m2)
{
   Situazione sit;
   byte g,m;
   minimi = new Vector();
   massimi = new Vector();

   try
   {
      FileInputStream f = new FileInputStream("situazione.dat");
      ObjectInputStream fIN = new ObjectInputStream(f);
```

```
      sit = (Situazione) fIN.readObject();
      g = sit.giorno;
      m = sit.mese;

      while (sit.precedente(g2, m2))
      {
        if (sit.successiva(g1, m1) || sit.stessoGiorno(g1,m1))
        {
          g = sit.giorno;
          m = sit.mese;
          Situazione min = new Situazione(sit);
          Situazione max = new Situazione(sit);
          while (sit.stessoGiorno(g,m))
          {
            min.impostaMinimi(sit);
            max.impostaMassimi(sit);
            try
            {
              sit = (Situazione) fIN.readObject();
            }
            catch(EOFException e)
            {
              minimi.addElement(min);
              massimi.addElement(max);
              return;
            }
          }
          minimi.addElement(min);
          massimi.addElement(max);
        }
        else
        {
          sit = (Situazione) fIN.readObject();
        }
      }
      f.close();
    }
    catch(Exception e)
    {
      return;
    }
  }
```

Al termine dell'esecuzione, i risultati si trovano memorizzati nei vettori *minimi* e *massimi* ed entrambi contengono lo stesso numero di elementi.

- **Metodo interr2**

```
public static void interr2(byte g, byte m)
{
   Situazione sit;
   giornata = new Vector();
   try
   {
      FileInputStream f = new FileInputStream("situazione.dat");
      ObjectInputStream fIN = new ObjectInputStream(f);

      sit = (Situazione) fIN.readObject();
      while (sit.precedente(g, m))
      {
         if (sit.stessoGiorno(g, m))
         {
            giornata.addElement(sit);
         }
         sit = (Situazione) fIN.readObject();
      }
   }
   catch(Exception e)
   {
      return;
   }
}
```

Dopo aver definito la classe che consente di eseguire le due interrogazioni, si deve costruire un'interfaccia grafica per richiamare i metodi *interr1* e *interr2* usando i parametri corretti. Si usano poi gli attributi *Interrogazioni.minimi*, *Interrogazioni.massimi* e *Interrogazioni.giornata* per mostrare i risultati.
Il seguente programma, usando dei valori fissi, richiama le interrogazioni e stampa i risultati.

PROGRAMMA JAVA (*Prove.java*)

```
import java.util.*;

class Prove
{
   public static void main(String argv[])
   {
      Vector minimi, massimi, giornata;
      Situazione sit;
      byte g,m,g1,m1,g2,m2;

      // interrogazione
   }
}
```

Nella zona dell'interrogazione si devono inserire le istruzioni per richiamare uno dei due tipi di interrogazione definiti nella classe *Interrogazioni*.
Per esempio, l'interrogazione seguente trova, per ogni giorno dal 1/1 al 3/1, i valori minimi e massimi della temperatura, pressione e umidità.

```
g1=1; m1=1; g2=3; m2=1;
Interrogazioni.interr1(g1,m1,g2,m2);
minimi = Interrogazioni.minimi;
massimi = Interrogazioni.massimi;
for(int i=0; i<minimi.size(); i++)
{
   sit = (Situazione) minimi.elementAt(i);
   System.out.println("\nGIORNO: "+sit.giorno+" - MESE: "+sit.mese);
   System.out.println("==Valori Minimi==");
   sit.stampa();
   sit = (Situazione) massimi.elementAt(i);
   System.out.println("==Valori Massimi==");
   sit.stampa();
}
```

L'interrogazione seguente, invece, mostra tutte le situazioni del giorno 2/1 indicando i valori ora per ora.

```
g=2; m=1;
Interrogazioni.interr2(g, m);
giornata = Interrogazioni.giornata;
sit = (Situazione) giornata.elementAt(0);
System.out.println("\nGIORNO: "+sit.giorno+" - MESE: "+sit.mese);
for(int i=0; i<giornata.size(); i++)
{
   sit = (Situazione) giornata.elementAt(i);
   System.out.println("ORA: "+sit.ora);
   sit.stampa();
}
```