Dipartimento di Informatica e Sistemistica
Antonio Ruberti

"Sapienza" Università di Roma

Esercitazione 4

Corso di Tecniche di programmazione

Laurea in Ingegneria Informatica

(Canale di Ingegneria delle Reti e dei Sistemi Informatici - Polo di Rieti)

Anno Accademico 2007/2008

Tutor: Ing. Diego Rughetti

Esercizio

Si vuole realizzare un'applicazione per la gestione di una rubrica telefonica.

I dati della rubrica sono registrati in un file chiamato rubrica.txt con il seguente formato:

- Cognome e nome

- Indirizzo

- numero di telefono

Nel file originario i nomi sono registrati in maniera disordinata, ovvero non sono in ordine alfabetico.

L'applicazione dovrà fornire le seguenti funzionalità:

- Aggiunta di un nominativo alla rubrica (nominativo, indirizzo, telefono)

- Dato un nominativo A visualizzare o restituire i dati relativi ad A

- Ordinamento dei nomi della rubrica

La funzionalità di ordinamento deve essere fornita attraverso differenti metodi, ognuno dei quali implementa uno degli algoritmi visti a lezione. Al termine dell'operazione di ordinamento stampare a video il numero di passi compiuti dall'algoritmo.

Fornire un main di prova per le classi realizzate.

Soluzione

```
public class Contatto extends Comparable{
        private String nominativo;
        private String indirizzo;
        private String telefono;
        public Contatto(){}
        public Contatto(String n, String i, String t){
                this.nominativo = n;
                this.indirizzo = i;
                this.telefono = t;
        }
        ... metodi set(….) e get()......
        public int compareTo(Object o) {
                String a1 = this.nominativo;
                String a2 = ((Contatto) o).getNominativo(); // il cast serve
                return a1.compareTo(a2) ;
        }

}
```

Soluzione (2)

```
import java.io.*;

public class GestioneRubrica{
        private String nomeFile;
        public GestioneRubrica(String s){
                this.nomeFile = s;
        }
        public void AggiungiContatto(Contatto c){
                try{

                        File f = new File(nomeFile);
                        FileOutputStream file = new FileOutputStream(f, true);
                        PrintStream ps = new PrintStream(file);
                        ps.println(c.getNominativo());
                        ps.println(c.getIndirizzo());
                        ps.println(c.getTelefono());
                        ps.close();
                        file.close();
                }catch(Exception e){
                        e.printStackTrace();
                }
        }
```

```
public Contatto getInfoContatto(String s) throws Exception{
                        Contatto c = new Contatto();
                        try{

                                File f = new File(nomeFile);
                                FileInputStream file = new FileInputStream(f);
                                BufferedReader br = new BufferedReader(file);
                                String temp;
                                do{
                                        temp = br.readLine();
                                        if(temp.equals(s){
                                                break;
                                        }
                                }while(temp != null);
                                if(temp == null){
                                        ps.close();
                                        file.close();
                                        throw new Exception("Nome non presente in rubrica!");
                                }else{
                                        c.setNominativo(temp);
                                        c.setIndirizzo(br.readLine());
                                        c.setTelefono(br.readLine());
                                }
                                ps.close();
                                file.close();
                                return c;
                        }catch(Exception e){
                                e.printStackTrace();
                        }
        }
```

Soluzione (4)

```
public void selectionSort(){
                try{
                        Vector contatti = new Vector();
                        File f = new File(nomeFile);
                        FileInputStream file = new FileInputStream(f);
                        BufferedReader br = new BufferedReader(file);
                        String temp = br.readLine();
                        Contatto c;
                        while(temp != null){
                                c = new Contatto();
                                c.setNominativo(temp);
                                c.setIndirizzo(br.readLine());
                                c.setTelefono(br.readLine());
                                contatti.add(c);
                                temp = br.readLine();
                        }
                        int n = contatti.size();
                        int jmin;
                        Contatto temp1, temp2;
```

Soluzione (5)

```
for (int i=0; i<n-1; i++) {
        // trova il piu' piccolo elemento da i a n-1
        jmin = i;
        for (int j=i+1; j<n; j++) {
                if ((contatti.elementAt(j).compareTo(contatti.elementAt(jmin)))<0)
                        jmin = j;
                }
                // scambia gli elementi i e jmin
                temp1 = (Contatto) contatti.getElementAt(jmin);
                temp2 = (Contatto) contatti.getElementAt(i);
                contatti.removeElementAt(jmin);
                contatti.insertElementAt(temp2, jmin);
                contatti.removeElementAt(i);
                contatti.insertElementAt(temp1, i);
        }
```

Soluzione (6)

```
                br.close();
                file.close();
                f.delete();
                f.createNewFile();
                FileOutputStream file = new FileOutputStream(f, true);
                PrintStream ps = new PrintStream(file);
                Enumeration e = contatti.elements();
                while(e.hasMoreElements()){
                        temp1 = (Contatto) e.nextElement();
                        ps.println(temp1.getNominativo());
                        ps.println(temp1.getIndirizzo());
                        ps.println(temp1.getTelefono());
                }
                br.close();
                file.close();
        }
}
```

Soluzione (7)

```
public void BubbleSort(){
        …… Lettura file e caricamento dati in vettore (come SelectionSort())

                int n = contatti.size();
                Contatto temp1, temp2;
                boolean ordinato = false;
                for (int i=0; i<n-1 && !ordinato; i++) {
                ordinato = true;
                for (int j=n-1; j>i; j--){
                        if ((contatti.elementAt(j-1).compareTo(contatti.elementAt(j))) > 0) {
                                temp1 = (Contatto) contatti.getElementAt(j-1);
                                temp2 = (Contatto) contatti.getElementAt(j);
                                contatti.removeElementAt(j-1);
                                contatti.insertElementAt(temp2, j-1);
                                contatti.removeElementAt(j);
                                contatti.insertElementAt(temp1, j);
                                ordinato=false;
                        }
                }
        }
        br.close();
        file.close();
        ….. Scrittura nuovo file (come SelectionSort())….
}
```

Soluzione (8)

```
public static void insertionSort() {
            ... lettura dati da file e caricamento in vettore contatti
      Enumeration e = contatti.elements();
      Contatto tmp;
      Contatto [] a = new Contatto(contatti.size());
      int i = 0;
      while(e.hasMoreElements()){
         a[i] = (Contatto) e.nextElement();
         i++;
      }
      for (int i = 1; i < a.length; i++) {
         // gli elementi tra 0 e i-1 sono gia' ordinati
         // inserisci l'elemento i tra gli elementi ordinati nella giusta posizione
      tmp = a[i];
      int j;
      for (j = i; j > 0 && (a[j-1].copareTo(tmp) > 0); j--){
         a[j] = a[j-1];
      }
      a[j] = tmp;
            .....salvataggio dell'array ordinato nel file della ribrica
 }
```

Soluzione (9)

```
public static void ordinaMergesort() {
    try{
        Vector contatti = new Vector();
        File f = new File(nomeFile);
        FileInputStream file = new FileInputStream(f);
        BufferedReader br = new BufferedReader(file);
        String temp = br.readLine();
        Contatto c;
        while(temp != null){
            c = new Contatto();
            c.setNominativo(temp);
            c.setIndirizzo(br.readLine());
            c.setTelefono(br.readLine());
            contatti.add(c);
            temp = br.readLine();
        }
        br.close();
        file.close();
```

```
Contatti[] v = new Contatti[vect.getSize()];
int k = 0;
Enumeration e = vect.elements();
while(e.hasMoreElements()){
        v[k] = (Contatto) e.nextElement();
        k++;
}
mergesort(v,0,v.lenght-1);
f.delete();
f.createNewFile();
FileInputStream file = new FileInputStream(f);
BufferedReader br = new BufferedReader(file);
int i = 0;
while(i < v.length){
        br.println(v[i].getNominativo());
        br.println(v[i].getIndirizzo());
        br.println(v[i].getTelefono());
        i++;
}
br.close();
file.close();
} catch(Exception e){e.printStackTrace()}
}
```

Soluzione (10)

```
private static void mergesort(Contatto[] v, int inf, int sup) {
        if (inf < sup) {
                int med = (inf+sup)/2;
                mergesort(v,inf,med);
                mergesort(v,med+1,sup);
                merge(v,inf,med,sup);
        }
 }
```

Soluzione (11)

```
private static void merge(Contatto[] v, int inf, int med, int sup) {
    Contatti[] a = new Contatti[sup-inf+1];
    int i1 = inf;
    int i2 = med+1;
    int i = 0;
    while ((i1 <= med)&&(i2 <= sup)) { // entrambi i vettori contengono elementi
       if (!(v[i1].compareTo(v[i2]) > 0){
          a[i] = v[i1];
          i1++;
          i++;
       }
       else {
          a[i] = v[i2];
          i2++;
          i++;
       }
    }
```

Soluzione (12)

```
if (i2 > sup) // e' finito prima il secondo pezzo del vettore
      for (int k = i1; k <= med; k++) {
        a[i] = v[k];
        i++;
      }
    else // e' finito prima il primo pezzo del vettore
      for (int k = i2; k <= sup; k++) {
        a[i] = v[k];
        i++;
      }
    // copiamo il vettore ausiliario nel vettore originario
    for(int k = 0; k < a.length; k++)
      v[inf+k] = a[k];
  }
```