Supervision 3 Work James Chapman

7.3

public class StudentScores {

private List<String> mStudents = new LinkedList<>();

private Map<String,Integer> mMapScore = new HashMap<>();

private void addStudent(String Name,int Score) {

mStudents.add(Name);

mMapScore.put(Name,Score);

}

private List<String> ListStudents() {

Collections.sort(mStudents);

List<String> output = new LinkedList<>();

for (int i= 0;i<mStudents.size();i++) {

output.add(mStudents.get(i));

}

return output;

}

private List<String> TopScore(int score) {

List<String> output = new LinkedList<>();

for (int i= 0;i<mStudents.size();i++) {

if(mMapScore.get(mStudents.get(i))>score) {

output.add(mStudents.get(i));

}

}

return output;

}

private int Median() {

List<Integer> TScores = new LinkedList<>();

for (int i= 0;i<mStudents.size();i++) {

TScores.add(mMapScore.get(mStudents.get(i)));

}

Collections.sort(TScores);

int size = TScores.size();

if (size%2==0) {

return ((TScores.get(size/2)+TScores.get((size/2)+1))/2);

} else {

return (TScores.get((size/2)+1));

}

}

}

7.6

public class FilePrinter {

private int[][] mArray;

public FilePrinter(String filename) throws Exception {

Reader r = new FileReader(filename);

BufferedReader b = new BufferedReader(r);

b.mark(1000);

String line = b.readLine();

int length = -1;

while ( line != null) {

length++;

line=b.readLine();

}

mArray = new int[length][1];

b.reset();

line=b.readLine();

for(int i = 0;i<=length;i++) {

String[] nline = line.split(",");

mArray[i][0] = Integer.parseInt(nline[0]);

mArray[i][1] = Integer.parseInt(nline[1]);

line=b.readLine();

}

}

public void print() {

java.util.Arrays.sort(mArray, new Comparator<int[]>() {

public int compare(int[] a1,int[] a2) {

if (a1[0] ==a2[0]) {

if (a1[1]>a2[1]) {return 1;}

else { return (-1);}

} else {

if (a1[1]>a2[1]) {return 1;}

else { return (-1);}

}

}

});

for (int i = 0;i<mArray.length;i++) {

System.out.println(mArray[i][0] + "," + mArray[i][1]);

}

}

}

The problem with the first solution was that there was no easy way of converting the person from say a Lecturer to professor. The only way to do this was create a new object of the required rank. This meant that all the information held in fields in the previous object would be lost. Whilst with the state solution, the fields would all stay the same, but the rank would be swapped and therefore the objects methods. So the person could have the same profile, but a different functionality due to its position