Hands-on Experiment # 8 : Worksheet

Section_____1_____ Date_____March 23, 2020_____

No more than 3 students per one submission of this worksheet.

Student ID _____6238197821_____Name__Witchayut Thongyoi_____

Student ID _____6238228621_____Name__Sopon Kongnithigarn_____

Student ID _____6238233721_____ Name__Adam Morgan_____

Sorting.

We are revisiting the processing of *score.csv* again. Now that we know how to use array, things should be much simpler. Also, this time, you are also given with a class called *StudentScore* which is designed so that all scores of a student with a specific ID are stored in a single object of *StudentScore*.

First, study, compile and run *TestStudentScore.java* so that you get some idea of how the *StudentScore* class can be used.

Explain what each statement in TestStudentScore.java does.

```
boolean withHeader = true;  // declare variable withHeader
    StudentScore s1 = new StudentScore("5630000021,10,10,9,8,7"); // create object s1 to
store data and split by ','
    s1.printScore(withHeader); // call method printScore with parameter withHeader = true

    withHeader = false; // declare variable withHeader= false
    StudentScore s2 = new // create object s2 to store data and split by
','StudentScore("5630000121,8,8,10,6,5");
    s2.printScore(withHeader); // call method printScore with parameter withHeader = false
```

Complete *ScoreSort.java*.

Include the screenshots below.

```java
public static void sortByTotal(StudentScore[] data) {
    selectionSort(data);
}

public static void listTop(StudentScore[] sortedScores, int n) {
    for (int i = 999; i > 999 - n; i--) {
        if (i == 999)
            sortedScores[i].printScore(true);
        else
            sortedScores[i].printScore(false);
    }
}

public static int findMaxIndex(StudentScore[] data, int lastIndex) {
    int maxIndex = 0;
    for (int i = 1; i <= lastIndex; i++) {
        if (data[i].getTotalScore() > data[maxIndex].getTotalScore()) {
            maxIndex = i;
        }
    }
    return maxIndex;
}

public static void swapData(StudentScore[] data, int a, int b) {
    StudentScore temp = data[a];
    data[a] = data[b];
    data[b] = temp;
}

public static void selectionSort(StudentScore[] data) {
    for (int lastIndex = data.length - 1; lastIndex > 0; lastIndex--) {
        int maxIndex = findMaxIndex(data, lastIndex);
        swapData(data, maxIndex, lastIndex);
    }
}
```

List all your source code here.

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

public class ScoreSort {
  // Do not change main().
  public static void main(String[] args) throws IOException {
    StudentScore[] scores = readScoreFile(); // Read score data from "score.csv" and store the
data in an array of
                          // StudentScore
    sortByTotal(scores);
    listTop(scores, 25);
  }

  // List neccessary methods here.
  // Do not change method headers.
  public static StudentScore[] readScoreFile() throws IOException {
    Scanner sc = new Scanner(new File("score.csv"));
    sc.useDelimiter("\\r\\n|,");
    sc.nextLine();
    StudentScore[] score = new StudentScore[1000];
    int n = 0;
    while (sc.hasNext()) {
      String data = sc.nextLine();
      score[n++] = new StudentScore(data);
    }
    return score;
  }

  public static void sortByTotal(StudentScore[] data) {
    selectionSort(data);
  }

  public static void listTop(StudentScore[] sortedScores, int n) {
    for (int i = 999; i > 999 - n; i--) {
      if (i == 999)
        sortedScores[i].printScore(true);
```

```java
        else
          sortedScores[i].printScore(false);
    }
  }


  public static int findMaxIndex(StudentScore[] data, int lastIndex) {
    int maxIndex = 0;
    for (int i = 1; i <= lastIndex; i++) {
      if (data[i].getTotalScore() > data[maxIndex].getTotalScore()) {
        maxIndex = i;
      }
    }
    return maxIndex;
  }


  public static void swapData(StudentScore[] data, int a, int b) {
    StudentScore temp = data[a];
    data[a] = data[b];
    data[b] = temp;
  }


  public static void selectionSort(StudentScore[] data) {
    for (int lastIndex = data.length - 1; lastIndex > 0; lastIndex--) {
      int maxIndex = findMaxIndex(data, lastIndex);
      swapData(data, maxIndex, lastIndex);
    }
  }

}
```