Module: Core Java

Session 23: Java IO practice

* This is a practice session; you will work IO assignments
* You can discuss your doubts with the trainer

**Assignments:**

**Assignment 1 – File IO Assignment**

**Objectives:**

The assignment is designed to develop and test your ability to write programs involving:

* Reading from files
* Writing into files

**Introduction**

The objective of this assignment is to write a program to process a set of student marks by reading them from an input file and writing them to an output file. The program should also write its output to the standard output

**Task Description:**

Each line of the input file follows this format: a name (which may be considered as a unique key for the purpose of this assignment) followed by a mark. An example is given below which lists marks for three students, Simon, Anna, and Edward. Each name will be a single word.

|  |
| --- |
| Simon 4  Anna 10  Simon 4  Anna 9  Anna 5  Edward 10 |

Once the input has been processed, the next step is to create the output. This should be listed in two ways and written to a single output file. The first list should be in alphabetic order, the second list should be in merit order. For each student in each list print the number of marks and the average. Each list should be preceded by a title.

After both lists have been printed, print the number of students, the average of the student averages (this is not the same as the average of all the marks processed), and the standard deviation of the student averages (i.e. in the example below, this would be the s.d. Of 10.0, 8.0, 4.0). Processing the above input data should produce the following output. All floating point numbers should be printed to one decimal place. When printing the merit order each line should begin with the rank in the merit order, as shown below. Take care to correctly handle cases where two or more students have the same average (they should be listed as having equal rank).

|  |
| --- |
| Alpha order  Anna 3 8.0  Edward 1 10.0  Simon 2 4.0  Merit order  1 Edward 1 10.0  2 Anna 3 8.0  3 Simon 2 4.0  Number of students: 3  Average student mark: 7.3  Standard deviation: 3.1 |

The filenames should be passed in as command-line arguments. If the output file already exists it should be overwritten. Your code should make appropriate use of Java Collection classes.

All these statistics can be calculated by storing all the numbers in appropriate collections and performing the calculations by accessing the collections. This works fine, but may run into problems for very large files. It can therefore also be desirable to calculate the statistics incrementally, which does not involve storing the individual numbers. Calculating the average in this way is straightforward, and you may have already done it for the ArgStats part of the previous assignment. The equation for the standard deviation (the sample standard deviation) is given below. Note that this involves summing the squares of the differences between each value and the mean. Therefore, this equation has to be rearranged in order to calculate it incrementally. (The population standard deviation has a 1/N term, this is the sample s.d. with a 1/(N-1) term.)

It is a good idea to define a Stat Summary interface class with the following methods:

|  |
| --- |
| public interface StatSummary {  void add(double x);  int n();  double average();  double sDev();  } |