Lab Assignment 01, Object-Oriented Programming, CSE 271, Spring 2020
Department of Computer Science and Engineering, Miami University

Warm-up Lab (Review and Getting started with Eclipse)

Objectives:

1. Getting started with Eclipse
2. Write a program to perform simple statistical analysis

Getting started with Eclipse:

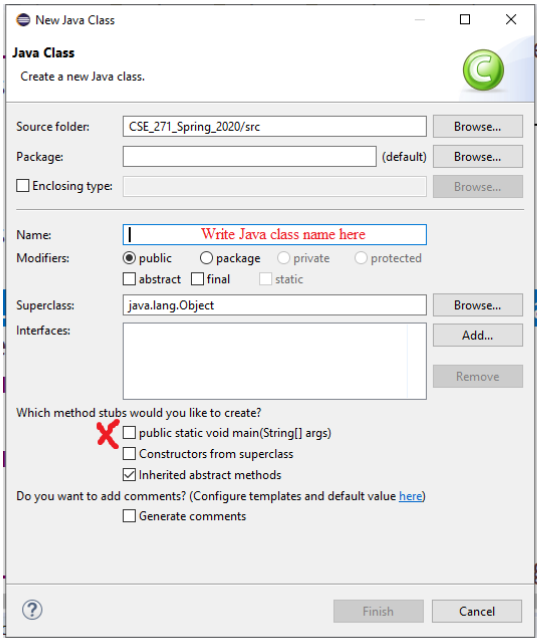
1. Run Eclipse. It will ask you to choose a workspace. Create a workspace in your universal disk space (m: drive).
   Check: Go to your m: drive and see where the workspace folder was created

2. Create a new Java project. Name the new project ***lab1***.
   Check: Go to your workspace (*Package Explorer*). If you don't see it then go to menu and select the following: "Window→Show View→Package Explorer ."
   There should now be a folder named, ***lab1*** in the package explorer. That folder contains several folders and files which you can see by expanding it (click on the "**>**" icon to expand it). It has a folder named, ***src*** which will contain our java (source) codes. Initially, it doesn't have any java files. We will later add our java code to this folder.

Now, write a program to perform simple statistical analysis:

A professor wants to perform some statistical calculations on his students' exam scores. Ask the professor about the number of students he has in the class, read the values and store these grades in an array, and print the statistical analysis of the grades.

1. Create a java class, **GradeStatistics** in the source (***src***) folder of the lab1 project. To do this, right click on the ***src*** folder and select "New→Class". It will open a window (see figure below) where you need to write the java class name. You can also select the check-box that says "public static void main(String[] args)".



Source folder: CSE_271_Spring_2020/src

Package: (default)

□ Enclosing type:

Name: Write Java class name here

Modifiers: ● public ○ package ○ private ○ protected

□ abstract □ final □ static

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

□ public static void main(String[] args)

□ Constructors from superclass

☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

□ Generate comments

2. Now, in the ***src*** folder you will see a java file names, **GradeStatistics.java**. Open it and write the following methods in it following the method descriptions:
   a. `public static double[] populateGrades()` – construct the array of type *double*, create Scanner class object to read in the number of students and their grades from the keyboard, store the grades in the array and then return that array.
   b. `public static void sortGrades(double grades[])` – it takes the grades array as an argument and sorts the values in ascending order (from low to high). You can use Arrays.sort() method to sort the array.
   c. `public static double getMean(double grades[])` – it takes the grades array as an argument, calculates the mean of all values and returns the result.
   d. `public static double getMedian(double grades[])` – it takes the grades array as an argument, calculates the median of all values and returns the result.
   e. `public static double[] getMinMax(double grades[])` – it takes the grades array as an argument, calculates the minimum and maximum of all values and returns the result in array. The result array has the minimum and maximum values in index 0 and 1, respectively.
   f. `public static void printGrades(double grades[])` – it takes the grades array as an argument, prints the content of the array from the left (index 0) to right. The grades are printed in the following format (separated by comma and a space).
   Format: 87.0, 89.0, 95.0, 97.0
   g. `public static void main(String args[])`– You need to call the methods you defined above to complete this method. Here is the order you must follow:
      1. Call populateGrades() method and store the returned array reference using a local array reference.
      2. Print the grades using the printGrades() method. You need to pass the array you populated using populateGrades() method.
      3. Sort the grades using sortGrades() method. You can update the local array reference to store sorted array reference. You can use the sorted array for rest of the tasks.
      4. Print the sorted grades using printGrades() method.
      5. Compute mean using the getMean() method and print it.
      6. Compute median using the getMedian() method and print it.
      7. Find minimum and maximum using getMinMax() method and print.

   Once you are done, select the menu: "Run→Run" to execute your program. Two sample runs are given below. Your program's output must follow the format shown in the sample runs.

Warm-up Lab (Review and Getting started with Eclipse)

Sample Runs: *(Inputs from the keyboard are shown in green color)*

```
Run1: (← this is not part of the output)
How many students do you have in the class: 4
Enter grade of Student 1: 10
Enter grade of Student 2: 100
Enter grade of Student 3: 20
Enter grade of Student 4: 30
The grades are: 10.0, 100.0, 20.0, 30.0
The sorted grades are: 10.0, 20.0, 30.0, 100.0
Mean = 40.0
Median = 25.0
Min = 10.0 Max = 100.0
Run2: (← this is not part of the output)
How many students do you have in the class: 5
Enter grade of Student 1: 100
Enter grade of Student 2: 90.5
Enter grade of Student 3: 95.5
Enter grade of Student 4: 10
Enter grade of Student 5: 20.75
The grades are: 100.0, 90.5, 95.5, 10.0, 20.75
The sorted grades are: 10.0, 20.75, 90.5, 95.5, 100.0
Mean = 63.35
Median = 90.5
Min = 10.0 Max = 100.0
```

Grading Rubrics:

Task	Grade
Create GradeStatistics class	2
Define populateGrades() method and work correctly	16
Define sortGrades() method and works correctly	10
Define getMean() method and works correctly	12
Define getMedian() method and works correctly	12
Define getMinMax() method and works correctly	12
Define printGrades() method and works correctly	12
Define main() – follows the order, correct format and output	22
Total	**100**

Submission Instructions:

- After you are done with the task, you need to show your work to one of the instructors present in the lab.
- Now, locate your source code **GradeStatistics.java** file at your workspace and submit it on CANVAS.