CS234 Computer Science II

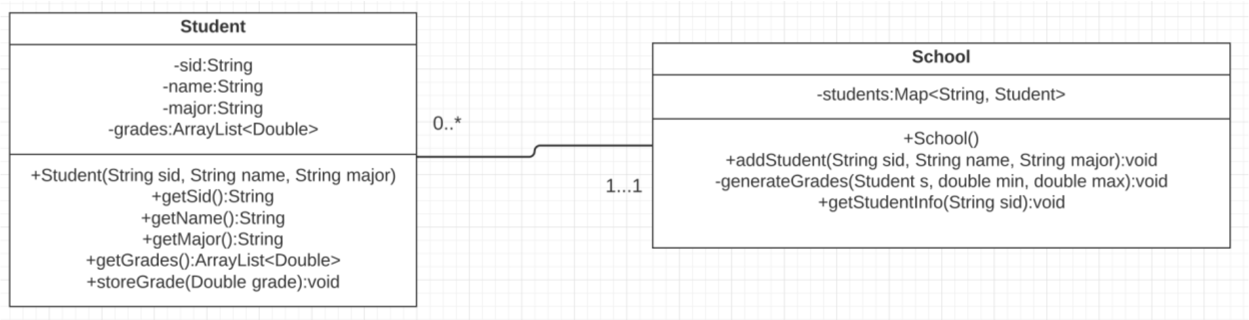
Lab 11
Total points: 100

Read the instructions carefully.

For this laboratory you need to create the definition of **two** Java **classes**.
You will practice:

- Understanding basic UML class diagrams (you will need this for your project)
- Use of Maps to store objects

For this lab you need to **create** the following **classes** (no main methods PLEASE!!)


Student
-sid:String
-name:String
-major:String
-grades:ArrayList<Double>
+Student(String sid, String name, String major)
+getSid():String
+getName():String
+getMajor():String
+getGrades():ArrayList<Double>
+storeGrade(Double grade):void
0..*
1...1
School
-students:Map<String, Student>
+School()
+addStudent(String sid, String name, String major):void
-generateGrades(Student s, double min, double max):void
+getStudentInfo(String sid):void

How many Java files?

Diagram elaborated in Lucidchart. It is free! www.lucidchart.com

Important information.

Student class

Student(). Constructor to initialize an object. It receives the student id, name, and major.

getSid(). Method to return the student's id.

getName(). Method to return the student's name.

getMajor(). Method to return the student's major.

getGrades(). Method to return the list of student's grades.

storeGrade(). Method to add a grade to the list of student's grades.

School class

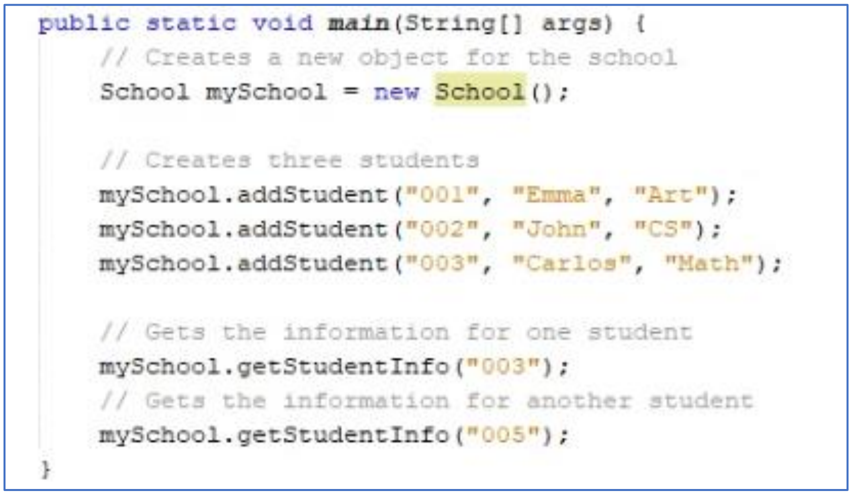
School(). Constructor to initialize the map

addStudent(). This method adds a **student** to the school. It instantiates the student object. It **uses** the *generateGrades()* method to generate random grades for the student.

generateGrades(). Private method that generates **five** random grades for a given student. The range of grades is between from 0 to 100.

getStudentInfo(). This method shows the information for a given ***student id***. It shows the id, name, major, the list of grades and the average of those grades. The average must be presented with two decimal positions at most. If the given student id is not in the school a message should be presented to the user. (See the examples below).

How to **test** your program?
You can implement a **tester program** like the following:



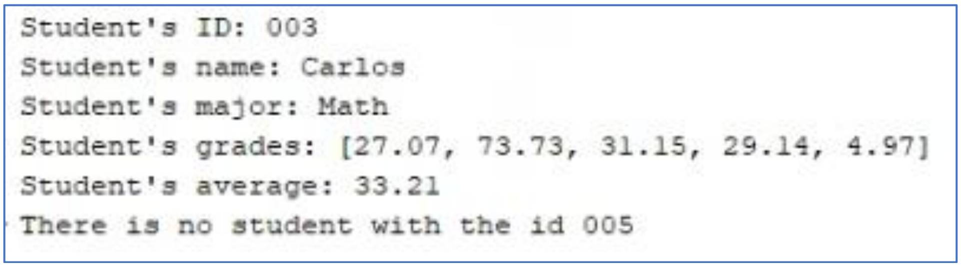
```java
public static void main(String[] args) {
    // Creates a new object for the school
    School mySchool = new School();

    // Creates three students
    mySchool.addStudent("001", "Emma", "Art");
    mySchool.addStudent("002", "John", "CS");
    mySchool.addStudent("003", "Carlos", "Math");

    // Gets the information for one student
    mySchool.getStudentInfo("003");
    // Gets the information for another student
    mySchool.getStudentInfo("005");
}
```

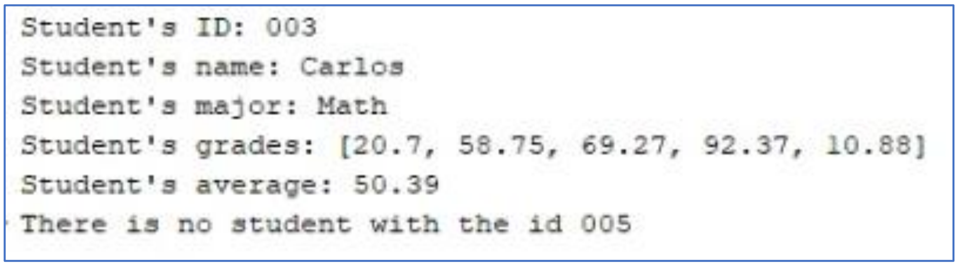
The **output** of the **tester** program is the following:

First run



```
Student's ID: 003
Student's name: Carlos
Student's major: Math
Student's grades: [27.07, 73.73, 31.15, 29.14, 4.97]
Student's average: 33.21
There is no student with the id 005
```

Second run



```
Student's ID: 003
Student's name: Carlos
Student's major: Math
Student's grades: [20.7, 58.75, 69.27, 92.37, 10.88]
Student's average: 50.39
There is no student with the id 005
```

Notice that the grades are randomly generated.

Please trace the tester program to **understand** the output.

Submission details:

Upload a **single ZIP** file.
Name your file as follows: **Lab11_Lastname_Firstname.zip**

Your **.zip** file must contain the following:

1. Your **.java** source files for your **class definitions** (.java files without the **main** method. No .class file). **Do not send the tester program**.
2. A **SINGLE PDF** with screenshots from your program running. Do not send .jpg files.

For this lab, you **do not need to submit the .txt** file with your instructions*****. **Why?** Because I will use my tester programs to use your classes. Therefore, it is **extremely important** that your **class and method names** are **the same** as they are shown in the class diagram and in my tester program.

***Note:** If you want to use packages, you need to submit the .txt file indicating how to use your package (i.e., how the path should be created)
In each .java file, **write** as a multiline comment at the beginning of the file the following:

1. Your name

The **zip** file must be uploaded to Canvas.
I do not accept image files; it must be a PDF file.

Make sure to check the **due date** for this activity on Canvas. Try to submit it before the due date so you can have time to check for improvements. **No late submissions. I do not accept solutions via email or as comments on Canvas.**

Make sure you are **submitting the correct files**. I will grade the files uploaded to Canvas.
Make sure you **test** your **classes** with a similar **tester program** as the ones I am showing in this lab (i.e., a .java program with a main method where you create objects from your class).

Use the *javac* and *java* commands with the tester program **before** submitting your solution. Compiling the tester program should **implicitly compile the other java files**. You don't compile them one by one. Just the tester program.

Make sure to review the **grading rubric.**

Read all the instructions carefully.

If you have questions, contact me before making assumptions about what you need to do for solving this assignment.

It is YOUR responsibility to contact me if something from these instructions is not clear or ambiguous.

Ensure that your code is original and developed by you.