EECS2030: LAB 1   Due: Sep 21st, 2021- 11:59 pm

The purpose of this lab is to

A) make sure that you have all the tools available for the rest of the term.

B) Write a simple code to practice automating the testing process.

C) Ensure that you know how to submit your programming assignments.

The first step is to download an editor in which you can write your java code. In this course we use *eclipse*, which is a professional editor for programming in many languages including java. In case you already have *eclipse* in your machine, you can skip step 1.

1. eclipse

To download, click on the link below where you can find the download links for different operating systems. Make sure that you click on a proper operating system.
https://www.eclipse.org/downloads/packages/release/luna/sr1/eclipse-ide-java-developers

The installation is easy as you only need to follow the instruction that is given by the installer. In case you have a problem installing it, please attend the lab and get help from the teaching assistants.

eclipse comes with the compiler and executer, so you don't need to install anything else.

2. Writing a simple code

A program in java should belong to a package, which in turn belongs to a project. Follow the steps below to write a java code:

1. Create a project by selecting File->new-> project.
2. Enter EECS2030 as the project name.
3. Click Finish and then select No.

Next step is to create a package.

1. Right click on EECS2030 project and select new->package.

2. Enter Lab1 and click on Finish. Remember, by default the name of the package should start with a capital letter.

Now, you are ready to write your program. For this

1. Right click on Lab1 package and select new->class
2. Enter `Lab` as the class name.

In this file and the class that is created you need to write two methods.

❖ The First Method

This method that is called `getMyID()`, accepts no parameter and returns your student number. The return type must be a string. Please make sure that you return a correct student number, otherwise we are not able to calculate your grade. The method signature should look like below:

```
public static String getMyID()
```

❖ The second Method

This method accepts a double number, which represents a grade and returns the letter grade equivalent of the input. The letter grade is computed using the following table:

Grade	Per Cent Range
A+	≥ 90
A	$80 \leq \; < 90$
B+	$75 \leq \; < 80$
B	$70 \leq \; < 75$
C+	$65 \leq \; < 70$
C	$60 \leq \; < 65$
D+	$55 \leq \; < 60$
D	$50 \leq \; < 55$
E	$45 \leq \; < 50$
F	< 45

The name of the method is `getLetterGrade()` and the return type is string. The method signature is:

```
public static String getLetterGrade(double)
```

Important Note: It is extremely important that the identifiers that you choose for the class and methods are the same as what is given in this description. Otherwise, our tester cannot read your code and you will lose the mark for this lab.

3. Testing your code

You should be able to use JUnit to test your code. Although we do not want you to submit your tester, we expect you to write your own tester, as this is part of the objective of this lab.

Since we assume that you have tested your code, using Junit, your code should not contain the `main` method.

When you write a tester, make sure that you choose a test case that is representative of each group (e.g. one test case when 90 < grade < 100, one test case when 80 < grade < 90 and so on). Also, you should have a test case per one borderline's value (e.g. 100, 90, 80, and so on).

4. Documentation

We do not grade your documentation, but you should get the habit of writing javaDoc for your code. So, please add a proper javaDoc for your code, where for the second method you add the following pre-condition:

```
@pre The input grade is a double number between zero and 100
inclusive.
```

5. Submit

You only submit one file that is called Lab.java via eClass by clicking on the lab link.