

The purpose of this ungraded lab is to:

- A. Ensure 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.

Although you can always use the Lassonde lab machine, even remotely, you can also download an editor for your computer at home to 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 on your machine, or you are going to use Lassonde lab's machine, then you can skip step 2.

1. Lab Policies

- a) **Academic Integrity:** Submit your own work. Do not copy code from classmates or online sources. All violations will be reported as academic misconduct.
- b) **Submission Format:** Submit only the file: Lab0.java. Do not upload ZIP files or project folders.
- c) **Work Environment:** Complete the three methods and test your submission by using tester file. Your code should not contain a main method for submission. Ensure your code compiles without errors.
- d) **Deadline:** Submit your .java file to the eClass course page by **Sunday, September 14 (11:59pm)**. No late submissions are accepted. Email submissions are not accepted.
- e) Your lab assignment is not graded during the weekly lab sessions scheduled. The lab sessions are meant to get your questions answered from TAs.

2. Set Up Java and Eclipse on Your Laptop

- a) **Install JDK 21:**
Download from <https://www.oracle.com/java/technologies/downloads/#java21> and follow the installation instructions for your OS.
- b) **Install Eclipse IDE:**
Visit <https://www.eclipse.org/downloads/>, download x86_64 installer and choose Eclipse IDE for Java Developers. During setup, make sure Eclipse uses JDK 21 as its runtime.

The installation is easy, as you only need to follow the instructions given by the installer. In case you encounter any issues during the installation, please attend the lab and seek help from the teaching assistants.

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

3. How to create a project in Eclipse? – Just for information

You've been given a zip folder that contains the actual project you'll be working on. The instructions provided below are not for this specific project, but rather to help you understand the overall working environment. They're meant as a reference in case you ever need to create a project on your own in the future.

A Java program 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_Lab0" as the project name.
3. Use an execution environment JRE – you can select the JavaSE-21 (anything above JavaSE-14 is good)
4. Uncheck Module
5. Click Finish.

The next step is to create a package.

1. Right-click on the src folder and select New -> Package.
2. Enter lab0 and click Finish. Remember, by convention, the name of the package should start with a lowercase letter.

Now, you are ready to write your program. Follow these steps:

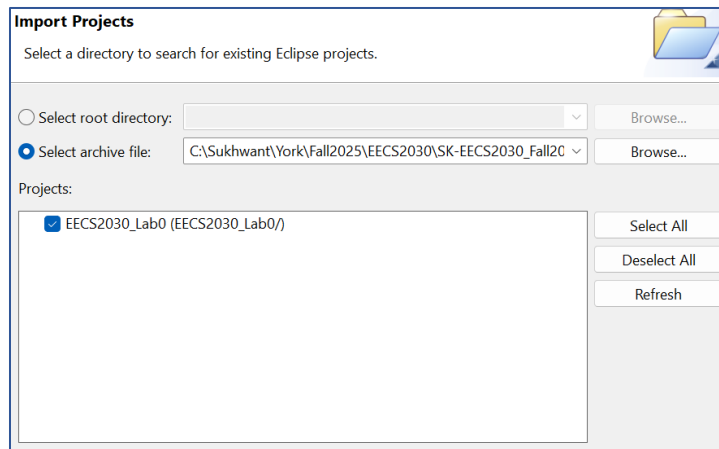
1. Right-click on the lab0 package and select New -> Class.
2. Enter Lab0 as the class name. Remember, by convention, the name of the class should start with an uppercase letter.
3. Don't select main method to be included as you are not writing main method in this lab.

In this file and the class that is created, you need to write a few methods.

4. Downloading and importing the Starter Project

Follow the below listed steps to set up your working environment by:

- a) Download the Eclipse Java project archive file from eClass: EECS2030_Lab0.zip
- b) Launch Eclipse and browse to EECS2030-workspace (for instance or your own created workspace).
- c) In Eclipse:
 - Choose File->Import
 - Under General, choose Existing Projects into workspace
 - Choose Select archive file. Browse your compressed zip folder and attach it.
 - Make sure that the EECS2030_Lab0 box is checked under Projects and you don't have the same project already in the workspace. Then Finish.



5. Programming Tasks for this Lab

In this lab, you need to write three methods in the Lab0 class.

❖ The First Method: `calculateBMI()`

This method calculates the Body Mass Index (BMI) given a person's weight in kilograms and height in meters. The BMI is calculated using the formula: $BMI = \text{weight} / (\text{height}^2)$.

The method should:

- Accept two parameters: weight (double) and height (double)
- Return the BMI as a double, rounded to one decimal place

Handle exceptional cases carefully:

- Return -1.0 if height is zero, negative, or extremely small (< 0.01)
- Return -1.0 if weight is negative
- Handle potential overflow/underflow situations for bmi, height and weight. Return -1.0.

[Look for `isInfinite()` and `isNaN()` methods in Java to handle exceptional cases]

Method Signature: `public static double calculateBMI(double weight, double height)`

Examples:

- Input: weight = 70.0, height = 1.75, Output: 22.9
- Input: weight = 85.0, height = 1.80, Output: 26.2
- Input: weight = 60.0, height = 0.0, Output: -1.0
- Input: weight = -5.0, height = 1.75, Output: -1.0

❖ The Second Method: findLongestWord()

This method takes an ArrayList of strings and returns the longest word. If there are multiple words with the same maximum length, return the first one encountered.

Handle exceptional cases carefully:

- Return an empty string ("") if the list is null or empty
- Handle null elements within the list by treating them as empty strings
- Handle lists containing only null elements
- Handle lists with mixed null and non-null elements

Method signature: public static String findLongestWord(ArrayList<String> words)

Examples:

- Input: {"apple", "banana", "cherry", "date"}, Output: "banana"
- Input: {"cat", "dog", "elephant", "bird"}, Output: "elephant"
- Input: {"hello", "world"}, Output: "hello" (both are same length, return first)
- Input: {}, Output: ""
- Input: {null, "hello", null, "world"}, Output: "hello"

❖ The Third Method: countVowels()

This method takes a string and counts the number of vowels (a, e, i, o, u) in it. The method should be case-insensitive, meaning both uppercase and lowercase vowels should be counted.

Handle exceptional cases carefully:

- Return 0 if the input string is null
- Handle empty strings appropriately
- Consider strings with special characters, numbers, and symbols
- Handle very long strings efficiently

Method signature: public static int countVowels(String text)

Examples:

- Input: "Hello World", Output: 3
- Input: "PROGRAMMING", Output: 3
- Input: "xyz", Output: 0
- Input: "AEIOUaeiou", Output: 10
- Input: "", Output: 0

- Input: null, Output: 0

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.

6. Documentation

Write proper JavaDoc documentation for each method.

Include:

- Method description
- Parameter descriptions using @param
- Return value description using @return
- Preconditions using @pre where appropriate

Generate the JavaDoc in the provided destination folder “Doc”, but only for your **Lab0.java** file. In the output, you should find an **index.html** file that includes the **Lab0 class** with all three methods properly documented.

Package lab0	
package lab0	
Classes	
Class	Description
Lab0	Lab0 class containing three methods for various computational tasks.

7. Testing your code

A test case file has been provided to evaluate your methods. Your implementation should pass all the test cases if it correctly handles both normal and edge cases. **Do not modify or update the test cases in any way.**

8. Submit

Submit only one file named "Lab0.java" via eClass by clicking on the lab link.

9. Marking Schema

This lab is not graded. However, for future labs, you are graded based on the correctness of your code. For each method, there will be a few test cases to examine the correctness of the code. All test cases carry the same weight.

Please note that in all the labs, even if the test cases are provided, we will test your code with a different set of test cases to ensure that you have tested your code completely.

Note:

- The program that does not compile will get zero marks.
- No resubmissions are allowed. Therefore, please make sure you submit the correct file within due date.
