## LAB 1 (WEEK 1)

**Topic:**

Programming and version control environment

**Instruction:**

* Follow all steps given in this lab.
* Answer all the questions below and present to demonstrator during lab.
* Combine all answers in one (1) file (Word or Pdf format). Submit the assignment in Putrablast before the lab session end (before end of Week 1) and to demonstrator.
* Refer lab rubric attached for assessment.

**Learning Outcome:**

At the end of this lab, student should be able to:

1. Compile and execute the sample program given (P2)
2. Understand to use versioning control (P3)

# USING ECLIPSE OPEN SOURCE TOOLS

Eclipse Open Source is a Java-based, open source development platform developed by IBM. It is simply a framework and a set of services for building applications from plug-in components. It provides an integrated development environment (IDE) for building Java programs. Editing, compiling, building, debugging and online help are integrated in one graphical user interface. You just need to enter your source code or open an existing source code in one window then click a button or a function key to compile the source code.

## DOWNLOADING AND INSTALLING ECLIPSE IDE

The installation of Eclipse is very straightforward. The guide to install and create your first Java Application using Eclipse IDE please refer to the video link in PutraBlast.

CREATING YOUR FIRST JAVA APPLICATION (USING WINDOWS)

# GETTING STARTED: YOUR FIRST JAVA APPLICATION

Checklist:

* The most recent version of Java SE Development Kit (JDK) – can be downloaded from <http://java.sun.com/javase/downloads/index.jsp>
* A text editor – such as Notepad or WordPad

Your first application, HelloWorld will simply display the greeting “Hello world!” using the command prompt. The steps are:



### Create a source file

**To create your first Java source file (or called as source code):**

* 1. Start any text editor then create a new document, if necessary.
  2. Type the class header **public class HelloWorld**. Press Enter once then type **{**.
  3. Press Enter again and then type **}**. We will add main()method between these curly braces.
  4. As shown in Figure 4 below, add the main()method between the curly braces by typing **public static void main (String[] args)**. Then type a set of curly braces for main().

public class HelloWorld

{

public static void main(String[] args)

{

}

}

**Figure 4** The main()method for the **HelloWorld** class

* 1. Next, add the statement **System.out.println(“Hello world!”)** within the main() method that can produce the output, Hello World! (Figure 5).

public class HelloWorld

{

public static void main(String[] args)

{

System.out.println("Hello world!"); // Display the string.

}

}

**Figure 5** Complete **HelloWorld** class

* 1. Save the application as **HelloWorld.java**.

**Important**: Make sure the file extension is .java by appending the .java to the file name when saving the file.

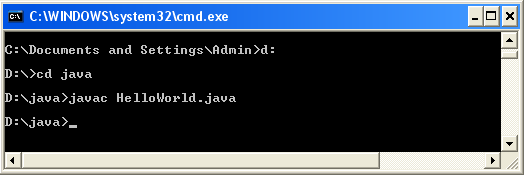
Now, you may exit the text editor.

### Compile the source file

**After saving the application, you must compile the source code into a .class file (or called as bytecode). To compile your source code from the command line:**

* 1. Start the command prompt.
  2. Change the default drive prompt to the drive and the directory where your application is stored.
  3. Type **javac** followed by the name of the file that contains the source code. Then press Enter.

For example, to compile the **HelloWorld.java**, you type **javac HelloWorld.java** then press Enter.



**Figure 6** Changing the directory of the application

**Important**: The name must match exactly, including the use of uppercase and lowercase characters. The compiler (javac) and launcher (java) tools are case-sensitive.

Now that you have a .class file, you can run the application.

### Run the application

**When the compile is successful, in order to run the source code from the command line:**

1. Type **java** followed by the class name of the source code at the command line and press Enter. For example, to run the **HelloWorld.java**, you type **java HelloWorld**.
2. The output should appear on the next line as shown in Figure 7 below.



**Figure 7** Output of the **HelloWorld** application

Congratulations, your program works!

**GITHUB**

GitHub consists of largest community of developers. It is useful for collaboration and communication between developers to work together on the same project by creating new versions of software without disrupting the current version. With GitHub, you can work on code with anyone from anywhere in the world.

STEPS TO USE GITHUB

1. Create a free GitHub account [here](https://github.com/join) for beginner stage. It only to get accessed to unlimited public repositories and only three collaborators for private repositories.

Please refer here:

<https://www.youtube.com/watch?v=LPT7v69guVY> <https://www.youtube.com/watch?v=HCeBd5GKNO8>

2. If you have face problem ‘Can't connect to any repository’, please refer to link:

<https://www.youtube.com/watch?v=rLEYw1caoBQ>

# *Instruction:*

# *Answer all the questions below and present to your demonstrator during lab.*

# *Combine all answers in one (1) file (Word or Pdf format). Submit the assignment in Putrablast before the lab session end (Week 1).*

1. Write, compile and test a class that prints your name on the screen. Save the class as **Name.java**.
2. Write, compile and test a class that prints your full name, matric number and your address in three separate lines on the screen. Save the class as **Student.java**.
3. Write a Java program that will prints multiple lines of text as follows:

Welcome

to

Java

Programming!

1. Making Money Bank intends to provide a new automated teller machine (ATM) for the customers to perform basic financial transactions. Each user has only one account at the bank. ATM users should be able to view their account balance, withdraw cash and deposit funds. The company has asked you to write a Java application for the ATM.

Write, compile and test a Java class that displays “Making Money Bank ATM Service” and “Welcome!” in an attractive layout on the console screen or in dialog box. Be certain to use appropriate comments in your class. Save the class as **MakingMoneyATM.java**.

Example of the layout you can create:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* Making Money Bank ATM Service \*

\* \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome!

**CSC3100 PROBLEM SOLVING AND PROGRAMMING CONCEPTS**

**LAB RUBRIC**

| **NO** | **CRITERIA** | **LEVEL 1** | **LEVEL 2** | **LEVEL 3** | **LEVEL 4** | **LEVEL 5** |
| --- | --- | --- | --- | --- | --- | --- |
| **Very Poor** | **Poor** | **Good** | **Very Good** | **Excellent** |
|  | Design  (Pre-programming phase) | Unable to construct | Able to construct but mistake on the symbol | Able to construct correctly | Able to construct correctly and use proper elements | Able to construct correctly, use proper elements and documentation |
|  | Functionality | Unable to construct a program | The program is producing incorrect results | The program produces correct results but does not display them correctly | The program works and produces the correct results and displays them correctly. It also meets most other specifications | The program works correctly and meets all specifications |
|  | Readability | Unable to organize the code | The code is poorly organized and very difficult to read | The code is readable only by someone who knows what it is supposed to be doing | The code is fairly easy to read | The code is exceptionally well organized and very easy to follow |
|  | Coding Skill Creativity | Student uses of code did not follow basic requirement. | Student uses of code follow a few basic requirement. | Student uses of follows most of the requirements but may contain a couple of errors. | Student uses of code follows all the requirements with no errors | Student’s use of code goes above and beyond, adding in additional challenges |
|  |  |  |  |  | Total | /20 |