

ENSF 593/594 – Principles of Software Development

Fall 2019

Lab Assignment 0 Part I: Introduction to Java

Due Dates
Before 11:59 on Friday September 13th

The objectives of this lab are to:

1. familiarize students with the process of writing, compiling and running a simple Java program.
2. Basic programming constructs


The following rules apply to this lab and all other lab assignments in future:

1. Before submitting your lab reports, take a moment to make sure that you are handing in all the material that is required. If you forget to hand something in, that is your fault; you can't use `I forgot' as an excuse to hand in parts of the assignment late.
2. **20% marks** will be deducted from the assignments handed in up to **24 hours** after each due date. It means if your mark is X out of Y, you will only gain 0.8 times X. There will be no credit for assignments turned in later than 24 hours after the due dates; they will be returned unmarked.

How to Compile and Run a Java Program

From Command-line: Once you edited your code using a text editor and saved it in your working directory (e.g `HelloWorld.java`), you can compile your Java program, using the following command on the command line in a terminal window:

```
javac HelloWorld.java
```

This command compiles your source code, links it with program libraries, and creates a byte-code file called: `HelloWorld.class`

To run your program, you should type at the command line: java HelloWorld

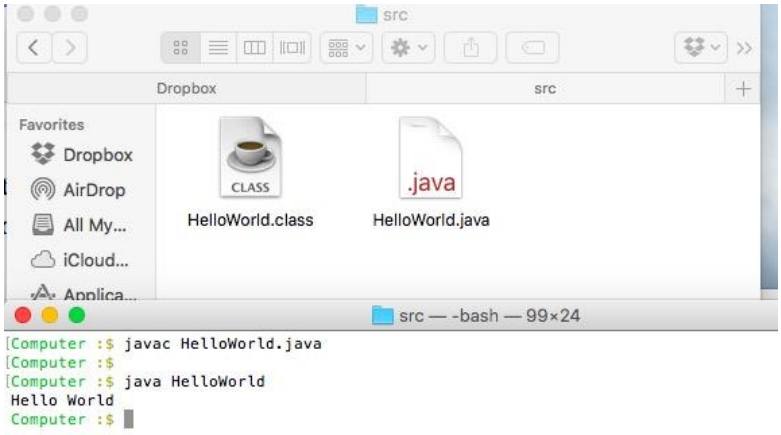


Figure 1-Running a program from command line

Eclipse IDE: In the Eclipse environment, you can run any Java class with a main method using the run command.

Run → Run As → Java Application

Figure 2-Running a program in Eclipse


Lab Exercise 1 – Simple if-else Structure

This is an in-lab exercise

Create the following logical expressions. The answer for the first question is given as an example.

Note: Assume all variables are integers.

a) x is greater than 100 or less than 0, and x is greater or equal to y

```
Solution:   (x  > 100 || x < 0 ) && x >= y
```

b) x is greater than y but less than z, and x can be either less than 10 or greater than or equal to 100.
c) a is less than or equal to 0 or greater than 90 and a is equal to b.
d) y is greater than or equal to 50. y must also be greater than z but less than or equal to x.

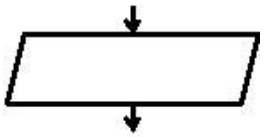
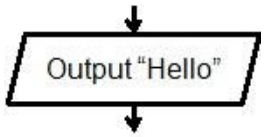
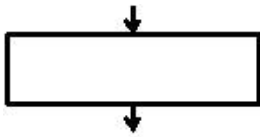
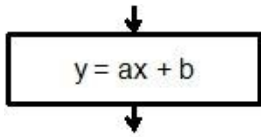
What to Submit:

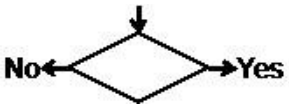
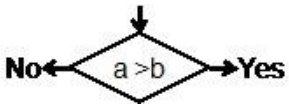
Your answers for questions a, b, c, and d.

Exercise 2: Another Flowchart

Flowcharts are often used to describe an algorithm visually. For the purpose of ENGG 233, we will only consider a few simple flowcharts "notations", as described in Table 1 below.

Table 1 – Flowchart Notations and Descriptions

Component	Description	Example
	This indicates the start and/or end of the algorithm. Every algorithm must have a start and end.	**See component**
	Input/Output. This is used to indicate that information is being output or being read in. It should be clear as to whether the component refers to input or output.	
	Statement or process. This describes a specific instruction (or group of similar instructions). A statement will have one or more entry points but only a single exit point.	


	Decision. Used to select an execution path based on a decision. There are one or more entry points, and two exit points. The exit points correspond to when the decision is true ("yes") or false ("no), and must be labeled appropriately.	

A lending institution wants to develop a questionnaire to determine what mortgage rate is available based on the type and duration of the mortgage desired. To this end, users enter their choice for (i) either an open or closed mortgage, and; (ii) the term duration (one year, three years or five years). Based on the user input, the available mortgage rate is provided based on the following table.

Term Duration	Open	Closed
1 yr	7.10 %	5.30 %
3 yr	7.50 %	5.00 %
5 yr	Not Available	5.75 %

You have been tasked with writing a program to implement the above. Specific details include (see above note regarding specifying implementation details)

- Read in the type of mortgage (i.e., open or closed).
- Use nested if-statements (i.e., the inner and outer if-statements should correspond to the term duration and type of mortgage, or *vice versa*).
- Please do the following

Draw a flowchart for an algorithm capable of solving this problem. The program should output appropriate messages if the following occur:

i. An invalid term duration is entered (e.g., three years)
ii. An invalid mortgage type is entered (e.g., "oppen" (spelling), "variable", "partially open")
iii. The input is correct but the option is unavailable (e.g, 5 yr open mortgage).

What to Submit:

Scan and submit your flowchart on D2L.



Exercise 3: Flowchart to code

This is an extended in-lab exercise

Implement your algorithm which you developed in previous exercise into a Java program. Test your program to make sure you can generate all six possible outputs in the table given in previous exercise (i.e. Exercise 2).

Your program can assume the input is correct. i.e. your program does NOT have to check for illegal input.

To get user input, you can use the following code:

```
import javax.swing.JOptionPane;
```

There is a function associated with JOptionPane, called: **showInputDialog**. This function creates a little dialog box on the screen that can prompt the user to enter an input. Then it returns the user's entry as a **string of characters.** For example, the following code segment is used to ask a user to enter the year that he/she was born, then it calculates the user's age and prints an output showing his/her age:

```
String userInput;
userInput = JOptionPane.showInputDialog("Which year were you
borne: ");
```

Then the user inputs can be used either as it is, or they be converted to a numeric value by using other available java parse functions. For example, to convert the user input to integer number, you may use the following statement:

```
int year  = Integer.parseInt(userInput);
```

What to Submit:

Submit your .java file on D2L