WESTERN UNIVERSITY
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

ES1036B
PROGRAMMING FUNDAMENTALS FOR ENGINEERS
WINTER 2022

Lab01 - Getting started & IntelliJ IDE

Instructor:
DR. QUAZI RAHMAN

Prepared by:
HIRA NADEEM,
HAREEM NISAR

Acknowledgement:
DR. A. OUDA

Lead TA: Hira Nadeem, hnadeem5@uwo.ca

---

1 Goal

In this lab you will learn how to use IntelliJ IDEA to create, edit, and run simple Java programs. Java and IntelliJ IDEA IDE are commonly used by developers both in academia and in industry, so your familiarity with them will be a valuable skill. It is recommended to use your own personal computer (laptop) to do this and the future labs.

2 Resources

Lecture notes:	Week 2 → "Unit 2: Java Fundamentals."
Pre-recorded video:	Week 1 → "Installing IntelliJ Idea IDE."
	Week 2 → "IntelliJ - Create a Project and Use Print Statements."

3 Deliverables

One zip file containing your project folder. Name it *username*_Lab1.zip. where username is the beginning part of your email (before @uwo.ca). Submission deadline: End of your lab session

Please note that your lab will be graded based on the following marks breakdown:

- 10%: comments + proper indentation + naming of identifiers
- 30%: Running code
- 60%: Demoing your understanding to your TA

4 Good Programming Practices

Below are several programming practices that should be followed to write and maintain quality codes. **Please note that you will be marked on all these components:**

- Include meaningful comments in your program. This will help you remember what each part of your code does, especially after long breaks from your work. Your TA will also appreciate understanding your code by going over your comments.
- Choose meaningful and descriptive names for your variables. There is a balance between descriptive names and code readability but always err on the side of descriptive.
- It is recommended that you follow the naming strategy for variables, methods and class names, as outlined in your course handout (Unit 2). Since the identifiers cannot contain white-space characters (spaces or tabs), words in an identifier should be separated by uppercase letters (myNewFunction(), myNewVariable). For class names, capitalize the first letter of each word in the name (e.g. MyClass, MatrixCalculator). For any constant name, use uppercase letters, and if needed, concatenate two or more words with underscore (e.g., MINIMUM_DRIVING_AGE)
- Initialize variables when declaring them. This means giving them initial values which are easier to track in your program if logical errors are present with your output.
- Indent and properly format your code. You should write your codes so that you can read and debug your code easily, and your teaching assistant can read and understand your code easily.
- **Include a header in each of your java class files**. The header should include your full name, UWO Student number, date the code was written and a brief description of the program in that file. Format below:

```
/*****************************************
* Add your full name here*
* Add your student number*
* Add Date*
* Give a brief description of the task *
*****************************************/

public class MyClass
{
    public static void main(String args[])
    {
      // Your code here
    }
}
```

5 Introduction to Java Programming

Java is primarily an object-oriented language. It means a Java program can be considered a collection of objects that communicate via calling each other's methods. A typical Java program includes a lot of classes, interfaces, objects, and other concepts from object-oriented programming. But you don't have to worry about that just yet.

Remember: Java program should have at least one class and the main() method inside it to start the program. The main() method is the entry point for any applications.

5.1 Java class

Before anything else, you create a java class. In your code, it will look something like this:

```
class MyFirstClass {

}
```

5.2 The declaration of the main method

The main() method is the entry point of any Java programs. It has a very specific syntax which you need to remember. Let's see an example of the simplest application that prints the text "Hello, Java" in the standard output:

```
class MyFirstClass {
  public static void main(String[] args) {
    System.out.println("Hello, Java");
  }
}
```

Here is a class named *MyFristClass*. The class contains the main() method for starting the program. It is important to mention that a class containing the main() method can have any name, but the main() method should always have the same name "main".

Let's take a closer look at the declaration of the main method:

```
  public static void main(String[] args) {
```

- The keyword ***public*** indicates that the method can be invoked from everywhere
- The keyword ***static*** indicates the method can be invoked without creating an instance of the class
- The keyword ***void*** indicates the method doesn't return any value
- The array variable ***args*** contains arguments entered at the command line. If there are no arguments then the array is empty.

As you can see, even the simplest Java application contains a lot of concepts. All of them will be studied during the feature lectures and lab exercises. For now, you just need to understand how to write and run a simple Java program with the main method. You will use these concepts in Question 1 of the assignment.

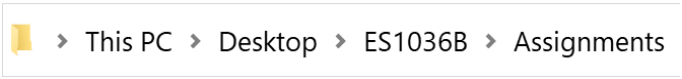
6 Getting Started

6.1 Install and Register the IntelliJ IDE

Follow instructions given under Unit 0 in your course handout in OWL "Installing IntelliJ Idea IDE"

6.2 Set up a Folder Structure

It is recommended that you dedicate a special folder called 'Assignments' inside your course folder **e.g.** This is where you will create and save all your project files.

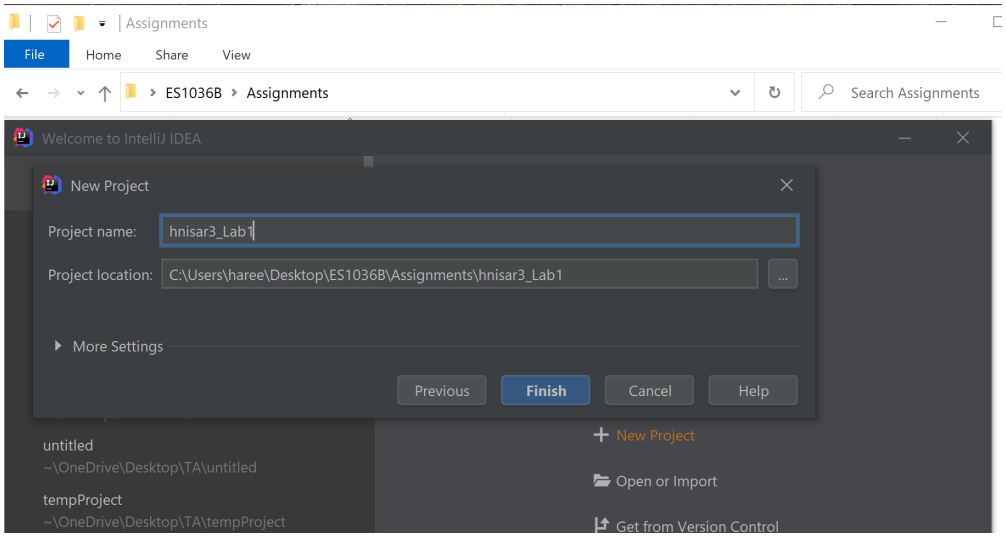


Note: You do not need to use this path, you can use your own path that you will remember. A good place is in your Documents folder.

While your project is open in IntelliJ Idea IDEA, you can view the path by looking next to your project name on the left-hand side where the file hierarchy is or by going to Edit > Copy Path/Reference.

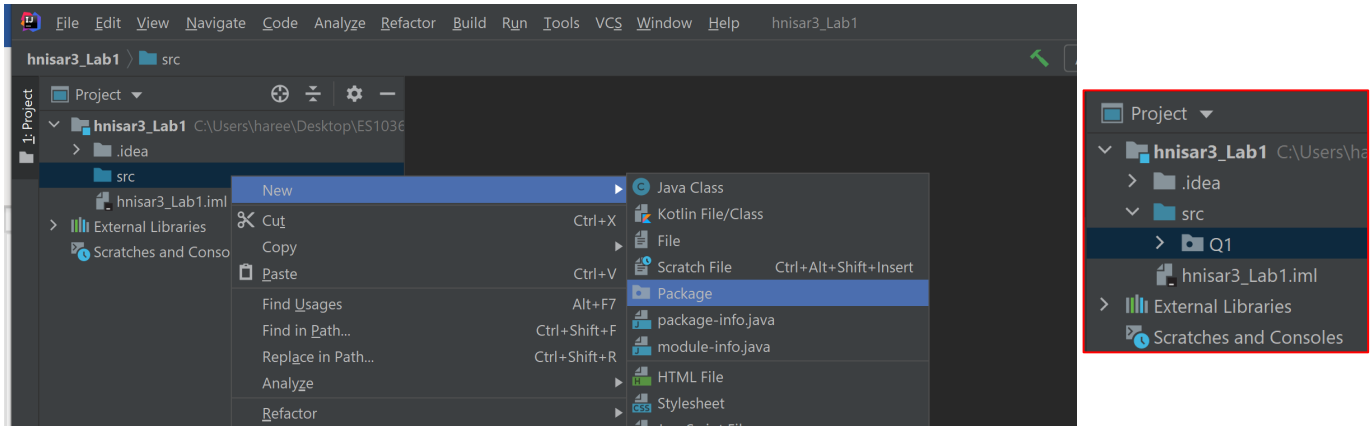
6.3 Create a Java Program

Step 1 – Create a Project
Follow instructions given under Week 2 → "IntelliJ - Create a Project and Use Print Statements" to see details. For each lab assignment, create a new project and follow the naming convention.
For example, for this assignment you will name the project *username_Lab1*



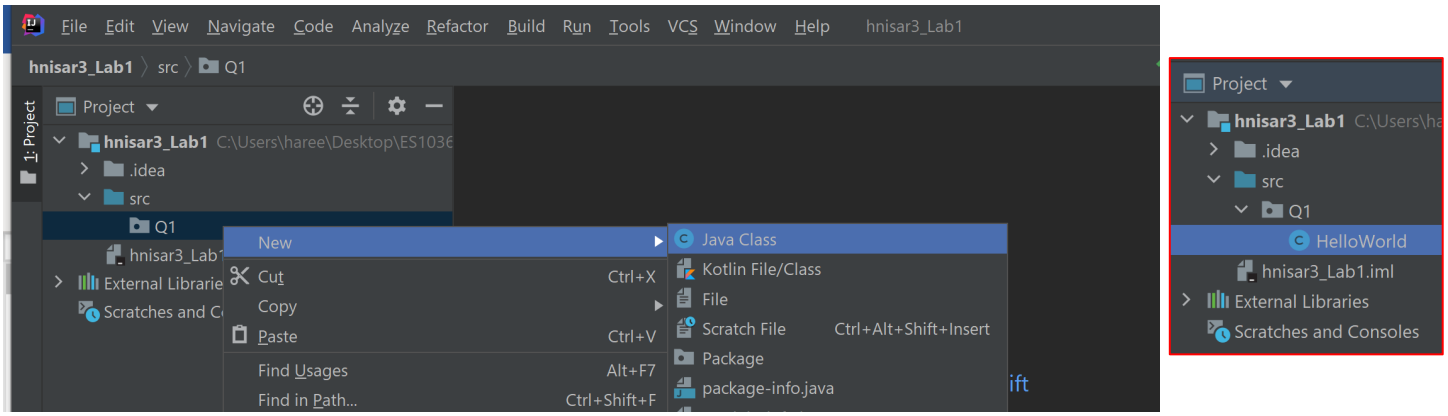
Step 2 – Create Packages

For each Question in the assignment create a new package. Right-click on *src* → New → Package. Name each package as Q1, Q2, and so on.
For example, for this assignment you will have three packages. Name them *Q1 Q2 Q3*



Step 3 – Add a Java Class

Inside your package, create a new java class where you will write your code. To add a java class to a package, right-click on the package e.g. Q0 → New → Java Class. Give your class an appropriate name according to the assignment instructions. In future labs, you will have multiple class files for one question.

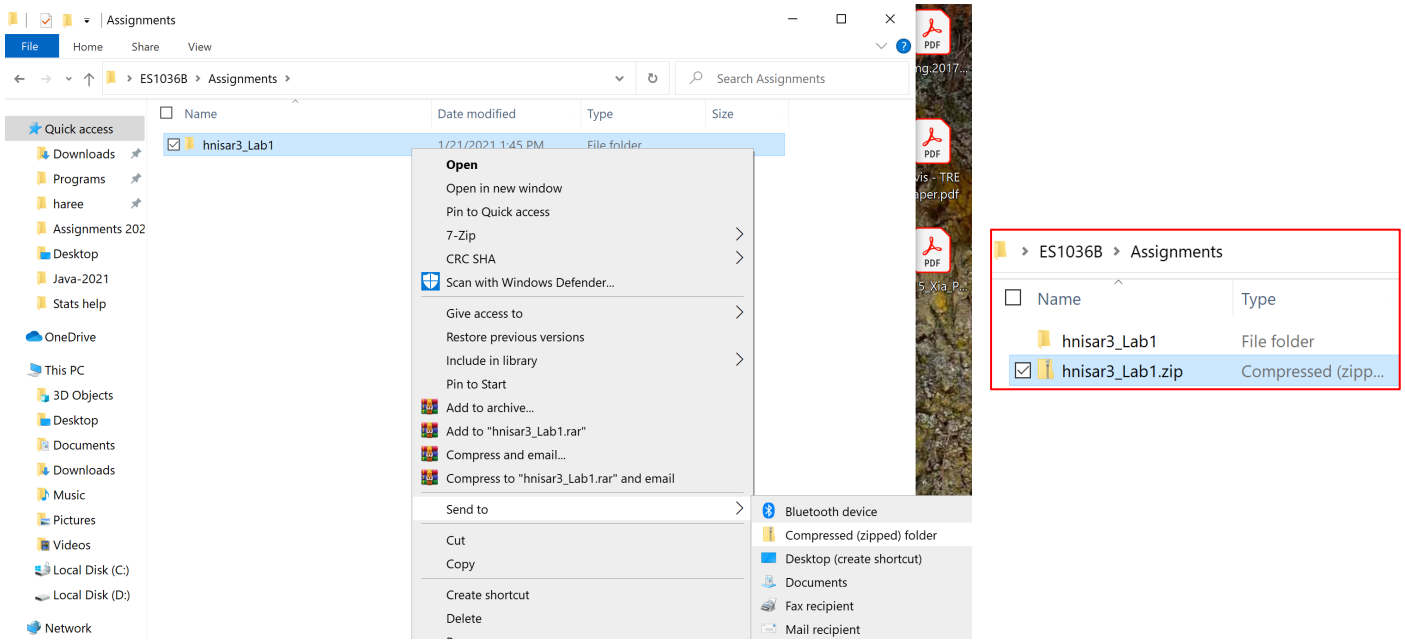


Step – 4 Code

Your files are now set. Start coding and have fun with it!

6.4 Zip File for Submission

Once the assignment is completed, go to your 'Assignments' folder. Select the project folder (e.g. *username_Lab1*). Right-click to select 'Send to' 'Compressed zipped folder'. Make sure it has the same naming convention (e.g. *username_Lab1.zip*). Upload this file to OWL as your submission .



7 Lab Assignment Questions

Before you begin working on your assignemnemt, make sure you understand and have implemented the instructions above in section 6 – *Getting Started.*

First, create a project named ***username*_Lab1** in IntelliJ IDEA.

7.1 Question 1 (5 Marks)

- Create a package named **Q1.**
- Create a java class named **HelloFromMe** which includes a commented header with your information: your full name, UWO Student number, date the code was written and a brief description of the program in that file (as described in Section 4: Good Coding Practice).
- The program should print the following text: "Hello, World! My name is *yourFirstName yourLastName*". Replace *yourFirstName* and *yourLastName* with your information as the example below:

```
Hello, World! My name is John Smith.
```

- Run the program to ensure that the message is printed out. Save your work. (Ctrl+S)

7.2 Question 2 (5 Marks)

- Create a package named **Q2.**
- Create a java class named **SimpleMath** which includes a commented header with your information.
- The program should print the answer of 5 divided by 2 in the following format:

```
The number 5 divided by 2 is: 2.5
```

- Is the output what you expect? If not, ask your TA for some help!
  Hint*: refer to Lecture Unit 2: Java Fundamentals Unit 2*
- Run your program to ensure the message prints out correctly. Save your work. (Ctrl+S).

7.3 Question 3 (10 marks)

- Create a package named **Q3.**
- Create a java class named **MyStudentID** which includes a commented header with your information.
- Declare an integer type variable called **myStudentNumber** and assign it the last four digits of your student number.
  If the last four digits of your student number starts with zero(s) (e.g., 0439), in that case, move the zero(s) to the end (0439 --> 4390), and use that number.
- Next, print the following:
  - The value stored in **myStudentNumber** using println() method.
  - The square-root of **myStudentNumber** by using the Math.sqrt() method.
  - The addition of **myStudentNumber** and the number 2022.
- For example, if the last four digits of your student ID are 1234, then the sample output will be:

```
The last 4 digits of my student number are: 1234
The square root of 1234 is: 35.12833614050059
The addition of 1234 and 2022 is: 3256
```

8 Submission

Zip the project file, name it ***username_Lab1.zip*** and submit in OWL by the end of your lab session.

If you have any questions about the Lab Handout, please contact the Lead TA, Hira Nadeem at hnadeem5@uwo.ca