**Assisted Practice: 2.1 Methods**

This section will guide you to:

* Create a Java project in your IDE
* Write a program in Java to create methods

This lab has three subsections, namely:

* + 1. Writing a program in Java to verify implementations of methods and ways of calling a method
    2. Executing the program and verifying working of methods
    3. Pushing the code to your GitHub repositories

**Step 2.1.1:** Writing a program in Java to verify implementations of methods and ways of calling a method

There are two ways you can perform this step; you can create a new Java project, or you can create a new Java class in the existing project. It is preferable to create a new Java class in the existing project but feel free to explore the first option. The steps mentioned below will work once you create a project in Java.

* Open Eclipse
* *[Right click]* on the **src** folder of the project
* Select *New* -> *Java Class* -> Enter the filename (follow camelCasing)
* Execute the code below resolving the warning and errors due to compatibility-related issues

//method demo

**package** methods;

**public** **class** MethodExecution {

**public** **int** multipynumbers(**int** a, **int** b) {

**int** z = a \* b;

**return** z;

}

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

MethodExecution b = **new** MethodExecution();

**int** ans = b.multipynumbers(10, 3);

System.***out***.println("Multipilcation is :" + ans);

}

}

**package** methods;

**public** **class** OverloadMethod {

//method overloading

**public** **void** area(**int** b, **int** h) {

System.***out***.println("Area of Triangle : " + (0.5 \* b \* h));

}

**public** **void** area(**int** r) {

System.***out***.println("Area of Circle : " + (3.14 \* r \* r));

}

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

OverloadMethod ob = **new** OverloadMethod();

ob.area(10, 12);

ob.area(5);

}

}

**package** methods;

**public** **class** CallMethod {

// call by value

**int** val = 150;

**int** operation(**int** val) {

val = val \* 10 / 100;

**return** (val);

}

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

CallMethod d = **new** CallMethod();

System.***out***.println("Before operation value of data is " + d.val);

d.operation(100);

System.***out***.println("After operation value of data is " + d.val);

}

}

**Step 2.1.2:** Executing the program and verify working of methods

Before you execute the program, check for syntactical corrections. If no errors are found, follow the steps mentioned below:

* ***[Right click]*** in the program space
* Select *Run As Java Application*







**Step 2.1.3:** Pushing the code to your GitHub repositories

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize your repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . -m “Changes have been committed.”**

* Push the files to the folder you initially created using the following command:

**git push -u origin master**