LAB 1 – Introduction to Java

Objectives:

At the end of this lab, students are able to:

i. To write simple Java Programs

ii. To display output on console Java

iii. To create, compile, and run Java program

iv. To display output using the JOptionPane output dialog boxes

1.1 Activity 1

1.1.1 Objective

To write a simple Java program using IDE TextPad.

1.1.2 Problem Description

You have been assigned to write a simple HelloWorld program using TextPad application. The program should be able to display a message "Hello World. This is my first time using Java".

[Estimated Time: 30 minutes]

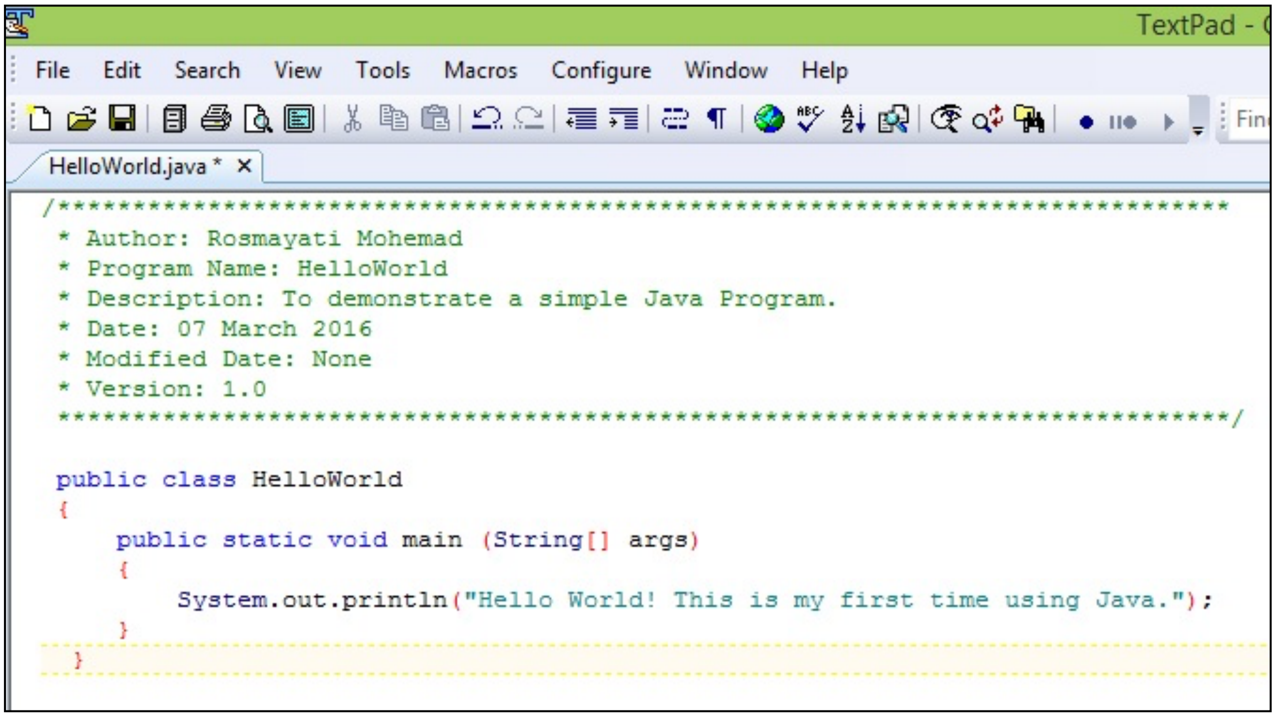
1.1.3 Solution Activity 1

Step 1:Go to CSF3102 –> Lab-Module1's folder and create Sub-working folder called Activity1. If the folders do not exist, create the folder first.

Step 2: Open IDE TextPad. Go to Start --> Search --> Type "TextPad".

Step 3: Create new file/class called HelloWorld.java save it into CSF3102\Lab-Module1\Activity1.

Step 4: Complete the coding as below.

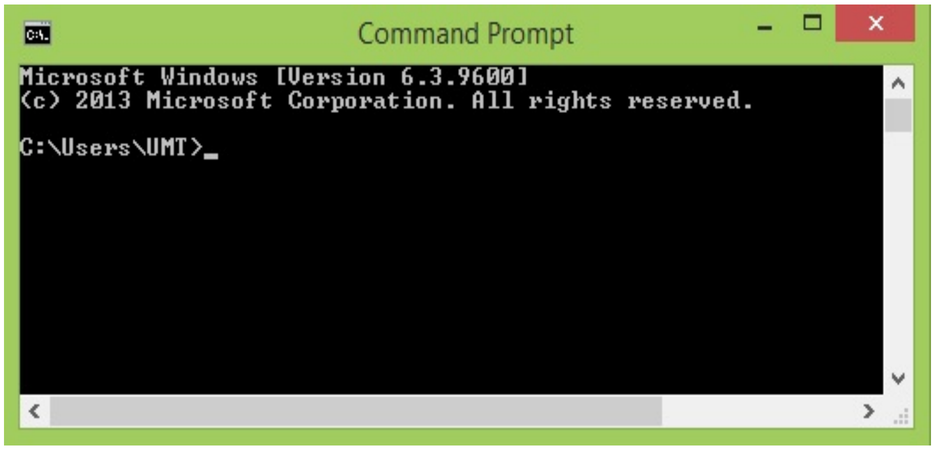


```
/************************************************************************
 * Author: Rosmayati Mohemad
 * Program Name: HelloWorld
 * Description: To demonstrate a simple Java Program.
 * Date: 07 March 2016
 * Modified Date: None
 * Version: 1.0
 ***********************************************************************/

 public class HelloWorld
 {
     public static void main (String[] args)
     {
         System.out.println("Hello World! This is my first time using Java.");
     }
  }
```

Step 5: Save the HelloWorld.java's file.

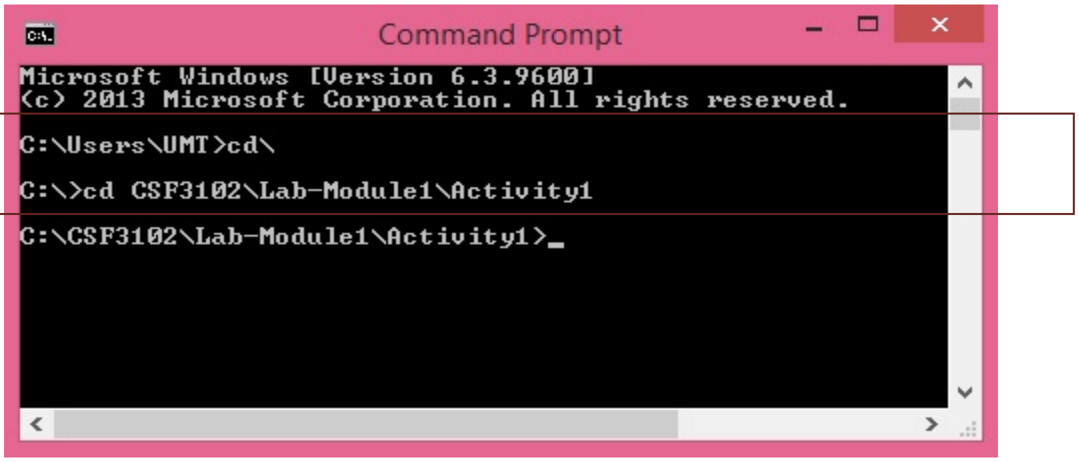
Step 6: Compile the program.

Step 6.1: Open Command Prompt. Go to Start --> Search --> Type "cmd". The Command Prompt window will appear as below



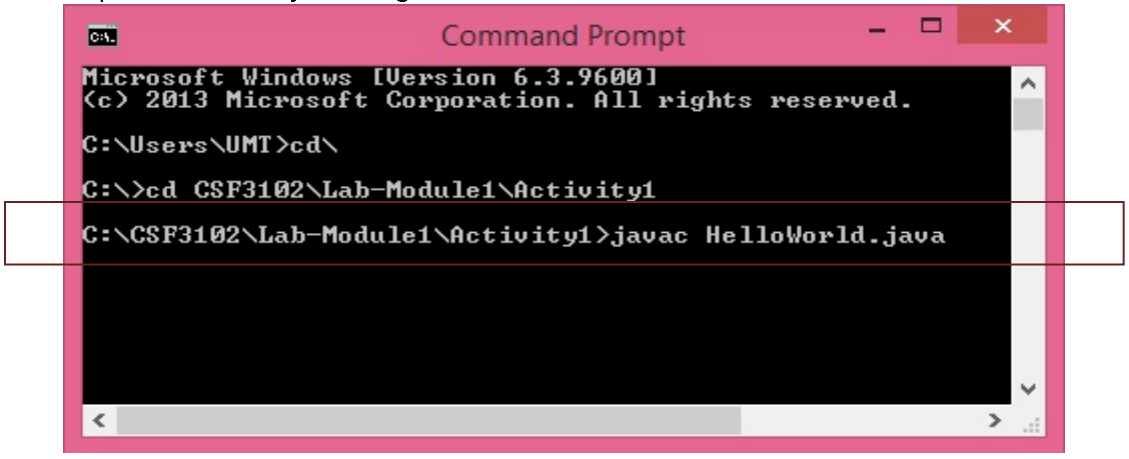
```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\UMT>_
```

Step 6.2: Go to the current folder where the HelloWorld.java is located using the command below.



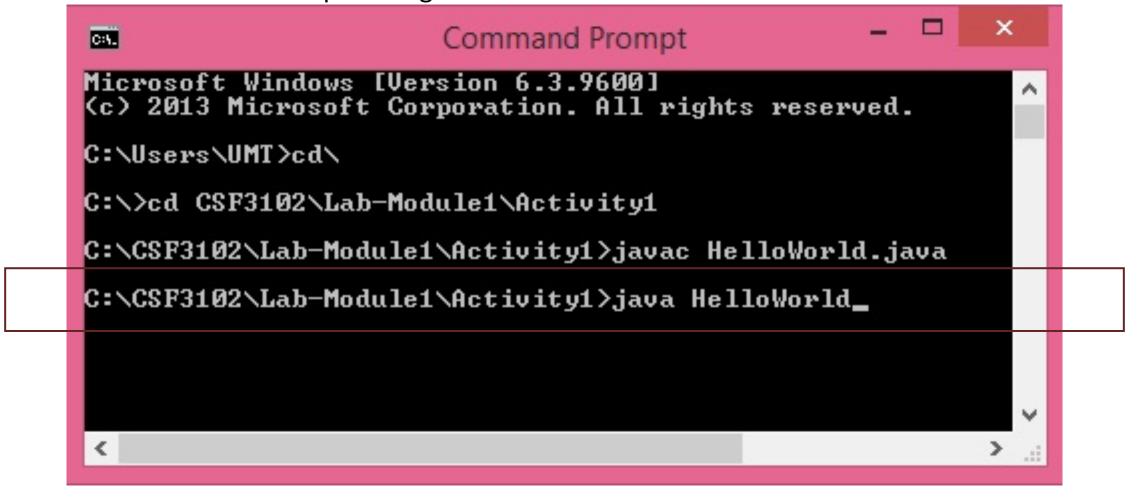
```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\UMT>cd\

C:\>cd CSF3102\Lab-Module1\Activity1

C:\CSF3102\Lab-Module1\Activity1>_
```

Step 6.3: Compile HelloWorld.java using the command below.



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\UMT>cd\

C:\>cd CSF3102\Lab-Module1\Activity1

C:\CSF3102\Lab-Module1\Activity1>javac HelloWorld.java
```

Step 7: Run and evaluate the output using the command below.



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\UMT>cd\

C:\>cd CSF3102\Lab-Module1\Activity1

C:\CSF3102\Lab-Module1\Activity1>javac HelloWorld.java

C:\CSF3102\Lab-Module1\Activity1>java HelloWorld_
```

1.2 Activity 2

1.2.1 Objective

To write a simple Java program using IDE NetBeans.

1.2.2 Problem Description

You have been assigned to write a simple WhoAmI program using NetBeans application. The program should be able to display a message "Muhammad Abdullah. UK Number: SS11111. Program of Computer Science (Software Engineering)".

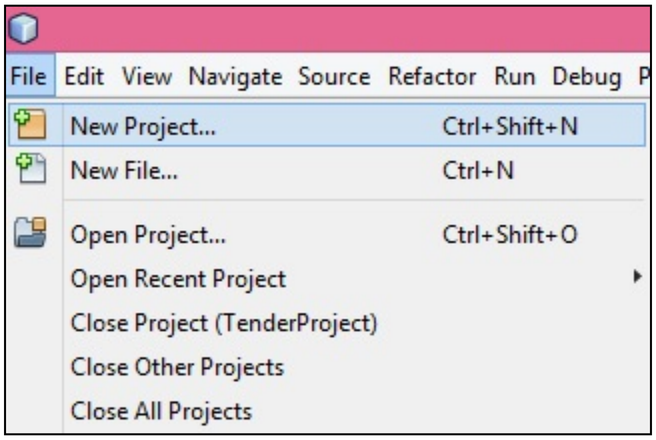
[Estimated Time: 30 minutes]

1.2.3 Solution Activity 2

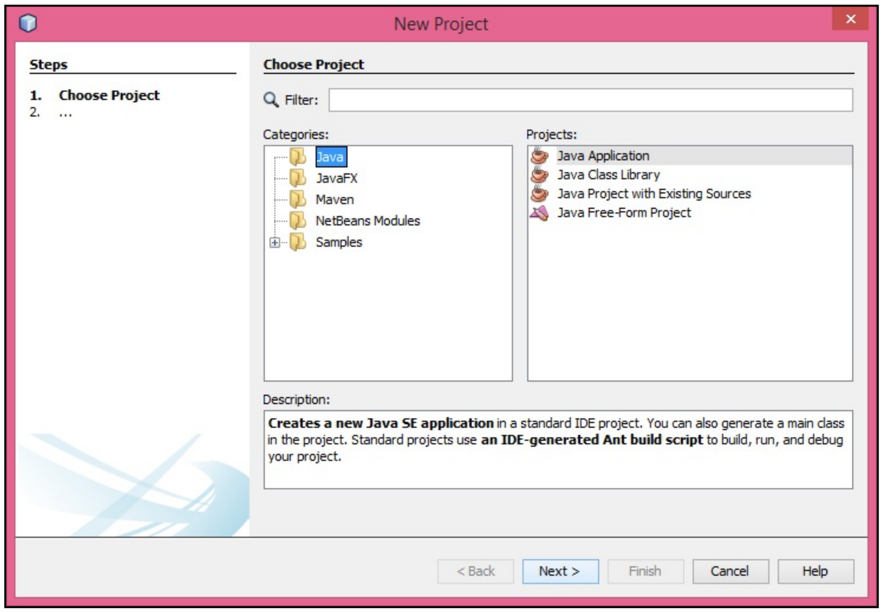
Step 1: Go to CSF3102 –> Lab-Module1's folder and create Sub-working folder called Activity2.

Step 2: Open IDE Netbeans.

Step 3: In the IDE, choose File -> New Project (Ctrl-Shift-N), as shown in the figure below.



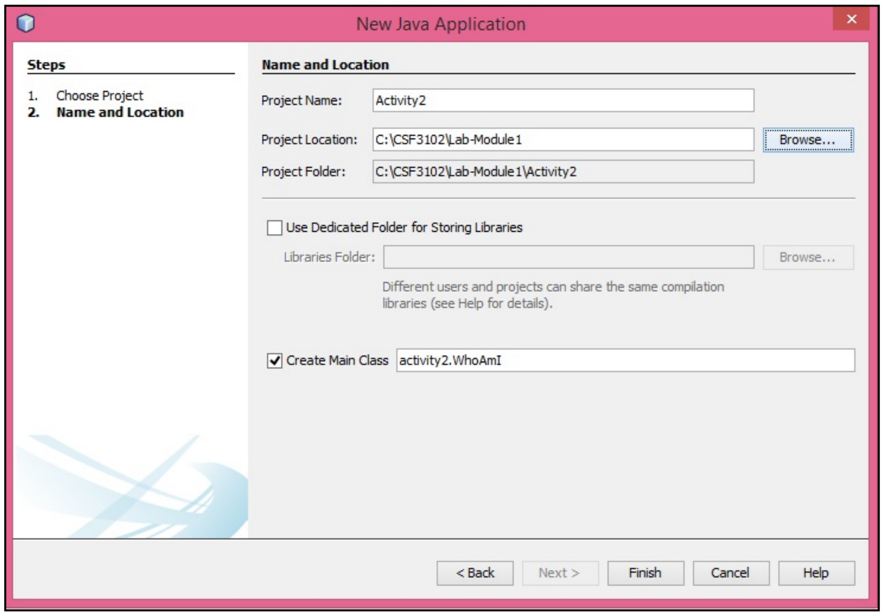
Step 4: In the New Project wizard, expand the Java category and select Java Application as shown in the figure below. Then click Next.



Description:
Creates a new Java SE application in a standard IDE project. You can also generate a main class in the project. Standard projects use **an IDE-generated Ant build script** to build, run, and debug your project.

Step 5: In the Name and Location page of the wizard, do the following (as shown in the figure below):

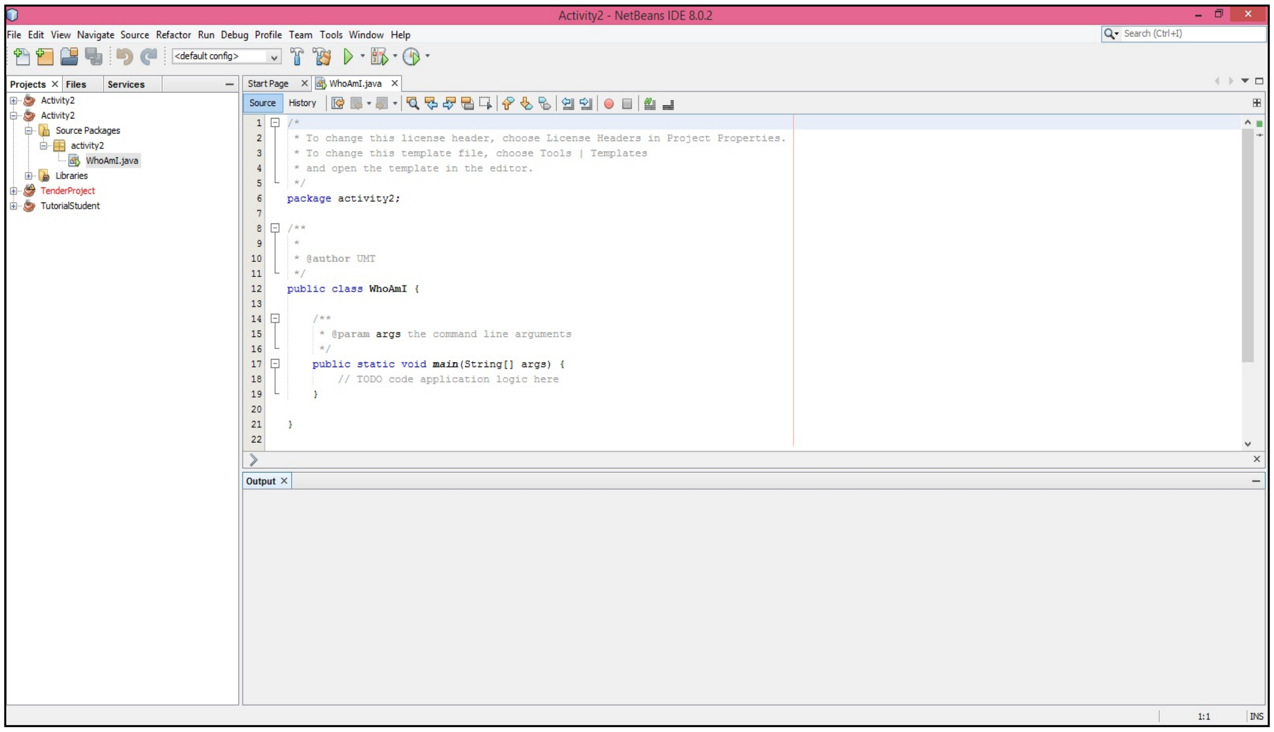
- In the Project Name field, type Activity2.
- Browse Project Location to CSF3102\Lab-Module1
- Leave the Use Dedicated Folder for Storing Libraries checkbox unselected.
- In the Create Main Class field, type activity2.WhoAmI.
- Leave the Set as Main Project checkbox selected.



Project Name: Activity2
Project Location: C:\CSF3102\Lab-Module1
Project Folder: C:\CSF3102\Lab-Module1\Activity2
☐ Use Dedicated Folder for Storing Libraries
Libraries Folder:
Different users and projects can share the same compilation libraries (see Help for details).
☑ Create Main Class activity2.WhoAmI

The project is created and opened in the IDE. You should see the following components:

- The Projects window, which contains a tree view of the components of the project, including source files, libraries that your code depends on, and so on.
- The Source Editor window with a file called WhoAmI.java is open.
- The Navigator window, which you can use to quickly navigate between elements within the selected class.
- The Output window, which lists compilation errors as well the output.



```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package activity2;

/**
 *
 * @author UMT
 */
public class WhoAmI {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
    }

}
```

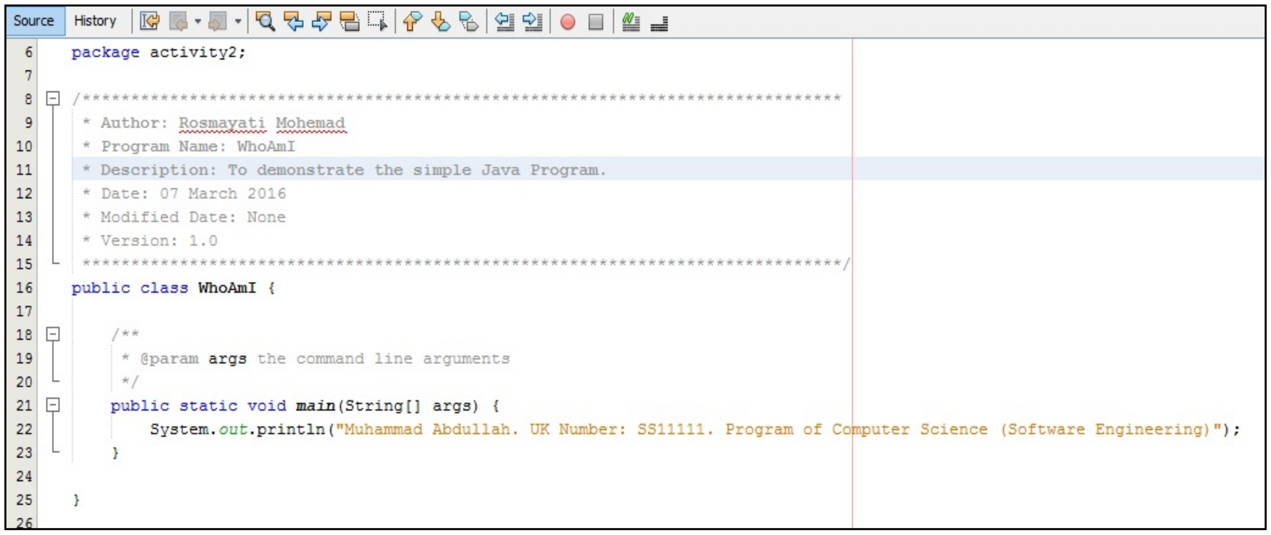
Step 6: Adding Code to the Generated Source File. Because you have selected the Create Main Class checkbox in the New Project wizard, the IDE has created a skeleton main class for you. You can add the "Muhammad Abdullah. UK Number: SS11111. Program of Computer Science (Software Engineering)" message to the skeleton code by replacing the line:

```
// TODO code application logic here
```

with the line:

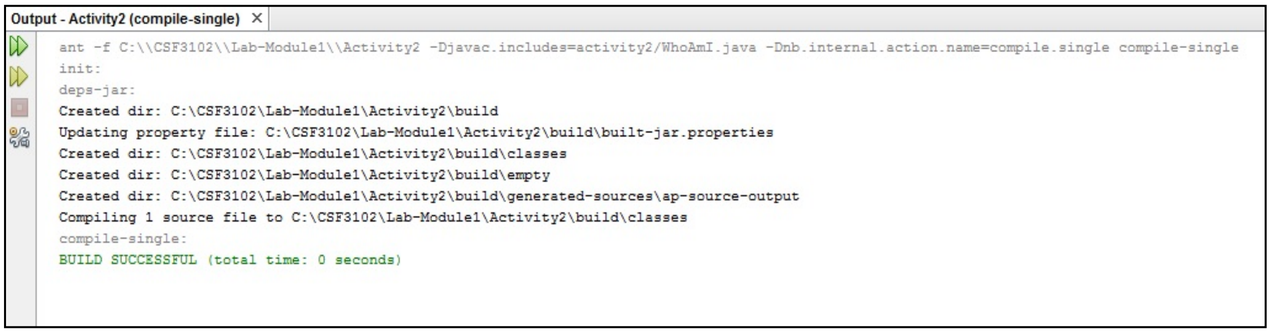
```
System.out.println("Muhammad Abdullah. UK Number: SS11111. Program of
Computer Science (Software Engineering)");
```

Step 7: Save the change by choosing File -> Save. The file should look something like the following code.



```
package activity2;

/*****************************************************************************
 * Author: Rosmayati Mohemad
 * Program Name: WhoAmI
 * Description: To demonstrate the simple Java Program.
 * Date: 07 March 2016
 * Modified Date: None
 * Version: 1.0
 *****************************************************************************/
public class WhoAmI {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        System.out.println("Muhammad Abdullah. UK Number: SS11111. Program of Computer Science (Software Engineering)");
    }

}
```

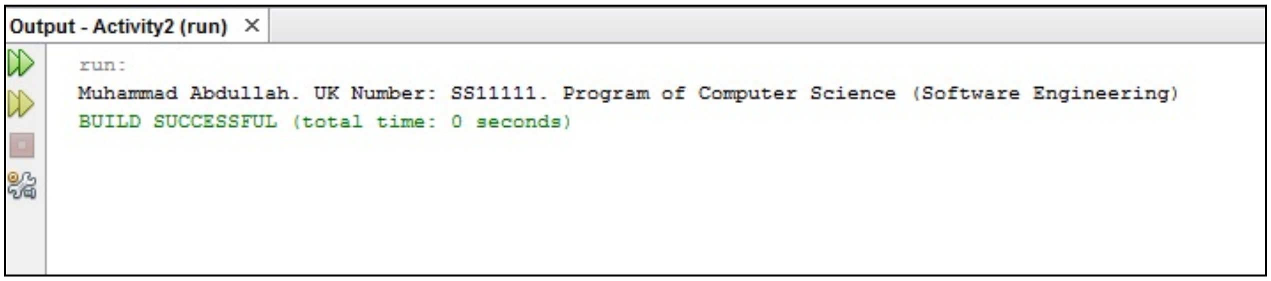
Step 8: Compiling the program. Go to Run -> Compile File (F9). Next, you should see the output window shows the compilation messages as below .



```
ant -f C:\\CSF3102\\Lab-Module1\\Activity2 -Djavac.includes=activity2/WhoAmI.java -Dnb.internal.action.name=compile.single compile-single
init:
deps-jar:
Created dir: C:\CSF3102\Lab-Module1\Activity2\build
Updating property file: C:\CSF3102\Lab-Module1\Activity2\build\built-jar.properties
Created dir: C:\CSF3102\Lab-Module1\Activity2\build\classes
Created dir: C:\CSF3102\Lab-Module1\Activity2\build\empty
Created dir: C:\CSF3102\Lab-Module1\Activity2\build\generated-sources\ap-source-output
Compiling 1 source file to C:\CSF3102\Lab-Module1\Activity2\build\classes
compile-single:
BUILD SUCCESSFUL (total time: 0 seconds)
```

If there are compilation errors, they are marked with red glyphs in the left and right margins of the Source Editor. The glyphs in the left margin indicate errors for the corresponding lines. The glyphs in the right margin show all of the areas of the file that have errors, including errors in lines that are not visible. You can mouse over an error mark to get a description of the error. You can click a glyph in the right margin to jump to the line with the error.

Step 9: Running the Program. Go to Run -> Run Project (F6). Next, you should see the output window displays the result as below.


Output - Activity2 (run)

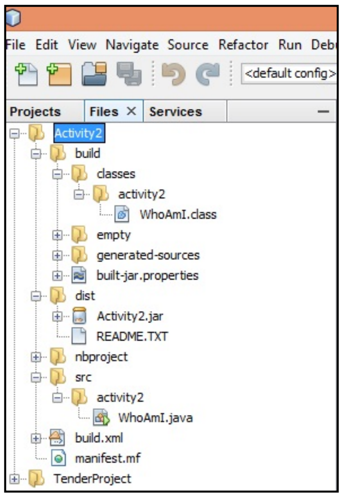
```
run:
Muhammad Abdullah. UK Number: SS11111. Program of Computer Science (Software Engineering)
BUILD SUCCESSFUL (total time: 0 seconds)
```

Step 10: Building and Deploying the Application. Choose Run > Clean and Build Main Project (Shift+F11)

Once you have written and test run your application, you can use the Clean and Build command to build your application for deployment. When you use the Clean and Build command, the IDE runs a build script that performs the following tasks:

- Deletes any previously compiled files and other build outputs.
- Recompiles the application and builds a JAR file containing the compiled files.

Step 11: You can view the build outputs by opening the Files window and expanding the Activity2 node. The compiled bytecode file WhoAmI.class is within the build/classes/activity2 sub node. A deployable JAR file that contains the WhoAmI.class is within the dist node.



```
Activity2
  build
    classes
      activity2
        WhoAmI.class
    empty
    generated-sources
    built-jar.properties
  dist
    Activity2.jar
    README.TXT
  nbproject
  src
    activity2
      WhoAmI.java
  build.xml
  manifest.mf
TenderProject
```

1.3 Activity 3

1.3.1 Objective

To write a simple program that can display multiple messages in several lines.

1.3.2 Problem Description

You have been assigned to write a simple MyProfile program using NetBeans application. The program should be able to display a message as shown below:

```
Resume

Name: Muhammad Abdullah.
Address: 1234, Jalan 5, Kampung 6, 78910, Kuala Terengganu, Terengganu
UK Number: SS11111.
Program: Program of Computer Science (Software Engineering)
```

[Estimated Time: 30 minutes]

1.4 Activity 4

1.4.1 Objective

To create, compile, run and examine a Java program.

1.4.2 Problem Description

You have been assigned to write a simple SelamatDatang program using NetBeans application. The program should be able to produce a message "Selamat Datang ke UMT". Based on the written program, answer the following questions:

i. Replace the `class` with `Class`. What happened? If error occurs, discuss the reason why the error occurs.
ii. Replace the `main` with `Main`. What happened? If error occurs, discuss the reason why the error occurs.
iii. Remove the semicolon at the end of the `println` statement. Explain what happened.

[Estimated Time: 30 minutes]

1.5 Activity 5

1.5.1 Objective

To examine, identify and fix the errors.

1.5.2 Problem Description

1. Identify and fix the errors in the following code. Write a FixErrors.java to fix the errors.

```
public class FixErrors
{
        Public static void main(string[] args)
        {
                System.out.print(Welcome!);

        }

}
```

2. What is wrong with the following program statement? Rewrite the statement in Test.java to fix the error.

```
System.out.println('To be or not to be, that is the question.');
```

3. What is the output produce if the following codes are added after the previous `println` statement as in question 2? Explain the result.

```
System.out.print ("Show me the answer.");
System.out.println("I'll give you the answer.");
```

[Estimated Time: 30 minutes]

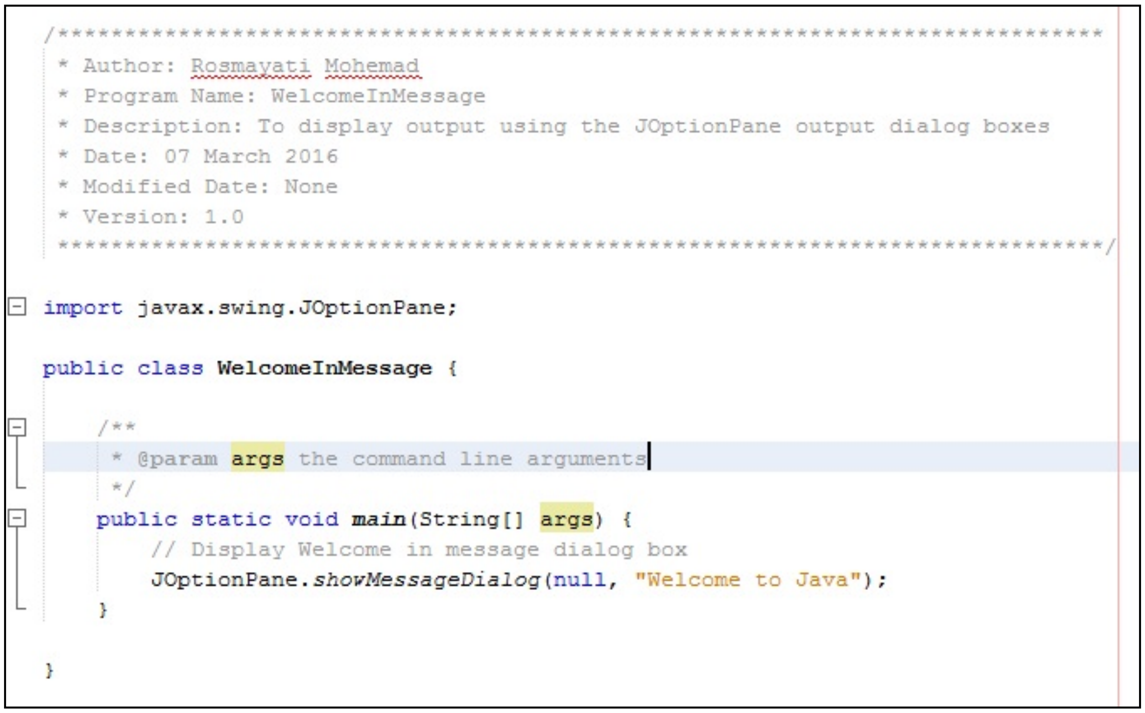
1.6 Activity 6

1.6.1 Objective

To display output using the JOptionPane output dialog boxes

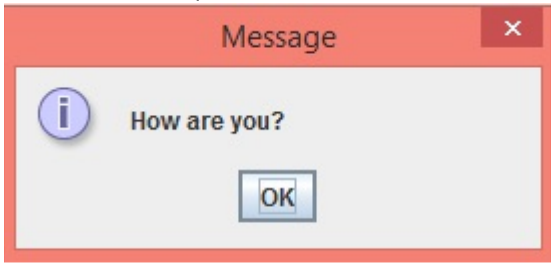
1.6.2 Problem Description

You have been assigned to write a simple WelcomeInMessage program using the JOptionPane output dialog boxes. The program should be able to display a message "Welcome to Java". The code is below.



```
/*****************************************************************************
 * Author: Rosmayati Mohemad
 * Program Name: WelcomeInMessage
 * Description: To display output using the JOptionPane output dialog boxes
 * Date: 07 March 2016
 * Modified Date: None
 * Version: 1.0
 *****************************************************************************/

import javax.swing.JOptionPane;

public class WelcomeInMessage {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // Display Welcome in message dialog box
        JOptionPane.showMessageDialog(null, "Welcome to Java");
    }

}
```

1. Compile and run WelcomeInMessageBox.java . What is the output of the code?
2. Rewrite the code that produces the output below:



Message
×
How are you?
OK