Advanced Programming 2015 – Year 2
Labwork 1: (5% - or 50 points out of 500 points for labwork this semester)

NOTE: ALL LABS TO BE COMPLETED IN PROJECTS USING ECLIPSE

Part 1 – Getting started with Eclipse projects (10 points)

Create an Eclipse Project called **Lab1Part1**. Create an 'EchoToWorld' java class within the Lab1Part1 project that executes and prints any string by calling a method called *sayHello(String str)* – where *str* is the String parameter to print to the default output device (System.out). Fully Javadoc the program including *@param* tag for the *sayHello(String str)* method.

Required activities and marking guideline:

- Create the Eclipse project Lab1Part1 (2 points)
- Write *sayHello(String str)* method (3 points)
- Add and generate the Javadoc (2 points)
- Jar and execute the program from the jar (3 points)

Part 2 – Multiple classes communicating in one project (10 points)

Create an Eclipse Project called **Lab1Part2**. Create three classes within this project which will communicate with each other through method calls. Put ALL three classes into a package called *cypher*. Call the first class Encoder.java and the second class Decoder.java and the third class should be called CodeBreaker.java. Add a method to the Encoder.java class which will receive a String and return and encoded version of that String using a method called encode(String str). Add a method to the Decoder.java class called decode(String str) which will receive a String (in coded form) and decode it and return the decoded String. Finally test the encoding and decoding of the Strings using the CodeBreaker class which should at least encode and decode a String using the methods written in the Encoder and decode using Decoder. Fully Javadoc each of the classes and methods in the classes.

Required activities and marking guideline:

- Implement the three classes in Project (3 points)
- Write the methods required (3 points)
- Add and generate Javadoc (2 points)
- Run the test class CodeBreaker (2 points)

Part 3 Multiple packages communicating in one project (10 points)

Create an Eclipse Project called **Lab1Part3**. Create two classes, within two separate packages within the project. Call the first class MathHelper.java and place it in a package called *mathematics*. Call the second class Application.java and place it in a package called *application*. Add a static method to the MathHelper.java class called multiplyNums(int x, int y) that receives two integers and returns the result of the two multiplied together. Add a main method to the Application.java class and call the MathHelper multiplyNums method with two test values. Finally, Jar this project and run it outside Eclipse using the jar file. Javadoc all classes and methods.

Required activities and marking guideline:

- Implement the two classes (2 points)
- Place the classes in their packages (2 points)
- Write the multiply method in MathHelper.java (2 points)
- Test MathHelper multiply method in Application.java (2 points)
- Jar and Javadoc (run the Jar file) (2 points)

Part 4 Putting it all together week 1 (Jars, Packages, IDE) (20 points)

Create an Eclipse Project called **Lab1Part4**. Create a simple JFrame GUI with a two JLabel's and two JTextField's and one JButton at the bottom. Call the frame CodeFrame.java. Set the text of the first JLabel to "Enter text to encode". Set the text of the second JLabel to "The encoded text is ". Add listeners and handlers so that when the user inputs the text to encode in the first JTextField and the JButton (labeled "Encode") is pushed the encoded text must be returned and displayed in the second JTextField (use any encoding system you desire). Place the GUI JFrame class in a package called *gui* and place the encoding class in a package called *encoder*. Finally Javadoc all classes and methods used, Jar the project and run it from the Jar file (note: you may re-use some of the code written in Part 2 above if you wish: Note: in fact if you Jar Part 2 of this lab you can import the jar including Encoder class using Project->Properties->Libraries->Add External Jar and call the encode method!).

Required activities and marking guideline:

- Create GUI with 2 labels and 2 fields and button (with layout) (5 points)
- Add listeners and handlers (4 points)
- Place classes in packages (4 points)
- Javadoc all of the project (3 points)
- Jar the project (include Javadoc in the jar) (2 points)
- Run the project from the Jarfile (2 points)