Advanced Programming 2015 – Year 2
Labwork 5: (6% - or 60 points out of 500 points for labwork this semester)

NOTE: ALL LABS TO BE COMPLETED IN PROJECTS USING ECLIPSE

NOTE: YOU MUST USE YOUR OWN EXAMPLES FOR THESE EXERCISES, I.E., YOU CANNOT RE-USE LECTURE EXAMPLE(S) AS SUBMISSIONS (THESE WILL BE GIVEN ZERO MARKS\POINTS)

REVISION EXERCISE 1 – (10 points) – THERE WILL BE 5 x 10 MARKS GOING TOWARD REVISION ACTIVITIES IN THE NEXT 5 LABS

Create a project called **Revision1**. In this project create a class called **Revision1**. This class should write a try-catch block to catch an **ArrayIndexOutOfBoundsException** from an attempt to access beyond the limits of a simple integer array, e.g., int[] array=new int[3]; array[3]=1;. Catch the array exception and send the message received (e.getMessage()) in the catch block to an external log file using the **Logger** utility class (java.util.Logger).

Required activities and marking guideline:

- Implement the try catch block for array (3 points)
- Implement the logging of the exception (4 points)
- Test program and verify contents of Logger file (3 points)

Part 1 – Use the 'extends Thread' approach for threading (15 points)

Create an Eclipse Project called **Lab5Part1**. Create a class called **ThreadWithExtends** that will print the letters "A", "B" and "C" (from a static array of Strings) to the default output device using a loop (repeated ten times at least). The **ThreadWithExtends** class must use threads to print the loop, use the **extends Thread** approach to add threads to this program. Create a second class called **TestThread**, which will test the creation and execution of the **ThreadWithExtends** program. Note that the output should be unpredictable, as the threads cannot guarantee the order of execution.

Required activities and marking guideline:

- Implement thread at class level using the extends approach (5 points)
- Implement the run method to loop through "A", "B", "C" (5 points)
- Write and run the TestThread application (5 points)

Part 2 – Use the 'implements Runnable' approach for threading (15 points)

Create an Eclipse Project called **Lab5Part2**. Create a class called **ThreadWithRunnable** that will print the integers 1, 2 and 3 (from a static array of integers) to the default output device using a loop (repeated ten times at least). The **ThreadWithRunnable** class must use threads to print the loop, use the **implements Runnable** approach to add threads to this program. Create a second class called **TestThread**, which will test the creation and execution of the **ThreadWithRunnable** program. Note that the output should be unpredictable, as the threads cannot guarantee the order of execution.

Required activities and marking guideline:

- Implement thread through implementation of Runnable interface (5 points)
- Fully implement the run method with looping output (5 points)
- Test the threaded application by running and starting the thread (5 points)

Part 3 – Synchronized (controlling thread access) (5 points)

Create an Eclipse Project called **Lab5Part3**. Fix **Part1** above so the threaded application ALWAYS prints A followed by B followed by C using a correctly placed synchronized block.

Required activities and marking guideline:

- Identified code needing restriction; fixed with synchronized block (3 points)
- Tested fix so that the output is always A, B, C (2 points)

Part 4 – A GUI Example that requires threads (15 points)

Create an Eclipse Project called **Lab5Part4**. Create a JFrame program called **StopTheLights** which draws a sequence of traffic lights to the screen in the usual pre-defined sequence of for traffic lights continuously, i.e., red, amber and green (use Graphics and drawOval and/or fillOval to draw the shapes\lights). Provide two buttons on the GUI to start the playing of the traffic lights sequence and one to stop the traffic light sequence. YOU MUST implement thread programming in the traffic lights painting sequence so that the GUI button(s) can be released to stop the lights.

Required activities and marking guideline:

- Code the Graphics lights sequence in a JFrame (red, amber, green) (5 points)
- Implement relevant listeners etc. (3 points)
- Implement the threads in the draw sequence (5 points)
- Successfully test\demonstrate the stopping of the light sequence (2 points)