Advanced Programming 2021 – Year 2
Labwork 5: (This lab is worth 60 points out of 500 points)

NOTE: ALL LABS TO BE COMPLETED IN PROJECTS USING ECLIPSE OR EQUIVALENT IDE (NO MORE TEXTPAD, EVER EVER!!!)

IMPORTANT NOTES:

- **NO COPYING PERMITTED AND ZERO MARKS WILL APPLY TO COPIED WORK. FURTHER ACTION MAY BE TAKEN AGAINST STUDENTS THAT HAVE BEEN FOUND TO COPY WORK.**

- **ASSESSMENT WILL INVOLVE ONE-TO-ONE QUESTIONS ABOUT YOUR SUBMITTED WORK. USE COMMENTS IN YOUR CODE TO ENSURE YOU DON'T FORGET WHY YOU WROTE CODE YOU MAY LATER BE ASKED ABOUT.**

- **ALL WORK MUST BE SUBMITTED TO MOODLE BY DATES SPECIFIED (2 LABS SUBMISSIONS OF FIVE LABS THROUGHOUT THE SEMESTER).**

- **MANY OF THE TASKS ASSIGNED BELOW CAN BE COMPLEX AND\OR THE DESCRIPTIONS MAY REQUIRE FURTHER CLARIFICATIONS. PLEASE USE THE AVAILABLE LAB TIMES TO ASK FOR CLARIFICATIONS AND ADVICE\HINTS ON THE TASKS BELOW.**

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 "X", "Y" and "Z" (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 "X", "Y", "Z" (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) (15 points)

Create an Eclipse Project called **Lab5Part3**. Add **TWO** new versions of **Part1** and **Part2** classes above so the threaded applications ALWAYS print X followed by Y followed by Z for **Part1**: and for **Part2** it always outputs 1 followed by 2 followed by 3 using a correctly placed **synchronized** blocks (Note: Because the operating system scheduling has ultimate control the behavior of the threads is not always 100% reliable on all platforms). Test both fixed versions.

Required activities and marking guideline:

- Identified code needing restriction; **fix** with **synchronized** block (7 points)
- Create the two new versions of **Part 1** and **Part 2** and test each (4 points)
- Test fixed classes so that the output is X, Y, Z and 1,2,3 in order (2 points)
- Javadoc the code and jar the project (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 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)