Lab 08 – Multithreading

1. Illustrate the concept of **Threads** using:

    a. **Runnable Interface**

    b. Extending **Thread** class

    Create 2 threads – main thread and a child thread.

2. Illustrate the concept of **Multithreading** using Runnable interface.

3. Create a class extending Thread class to print a multiplication table of a number supplied as a parameter. Create another class Tables which instantiates two objects of the above class to print multiplication table of 5 and 7.

4. Write an execute a JAVA program to create and initialize a matrix of integers. Create **n** threads by implementing Runnable interface (n:= number of rows in the matrix). Each of these threads computes a distinct row sum, which will be totaled by the main thread in the main class as the final sum of all elements in the matrix.

5. Write and execute a JAVA program to implement a **Producer/Consumer** problem using **inter-thread communication**.

6. Write and execute a JAVA program to illustrate **thread priority**.

7. Create 4 Threads and a main thread:

    a. First calculates row sum

    b. Second calculates column sum

    c. Third calculates principal diagonal sum

    d. Fourth calculates secondary diagonal sum

    e. Fifth checks for the uniqueness of the matrix elements

f. Main thread reads a square matrix from keyboard and displays whether the given matrix is magic square or not by obtaining the required data from sub threads.

8. Create a **Counter** class with the following instructions:

    a. Instance variables – private count

    b. Methods

        i. Synchronized void increment()

            1. Increments by 1 till it becomes 3 which is the maximum

            2. If count==3 then wait() till the count<3 before incrementing

            3. Wait() posts a message – waiting for decrement

        ii. Synchronized void decrement()

            1. Decrements by 1 till it becomes 0 which is the minimum

            2. If count==0 then wait() till count>0 before decrementing

            3. Wait() posts a message – waiting for increment

    c. **CounterThread** class

        i. Thread argument – increment or decrement

        ii. Calls the Counter's increment() / decrement() 20 times

        iii. Sleep time between the increment / decrement – random(0,500ms)