Lab Assignments – VII
MCA Semester III
CG and Java Lab (CS3307)

1. Create a multithreaded program that will add from 1 to some previously fixed integer *n.* Your program must create 8 threads to perform the task, and it should be versatile enough to add perfectly even when *n* is not divisible by 8. [Divide *n* into 8 equal ranges, assign them to each thread, and for any remaining last few numbers, perform the addition in main thread]

2. Create a stack with maximum *n* elements, where *n* is the array size you need to consider in the implementation of stack, and will be passed via a constructor, i.e., "*MyStack s = new MyStack(10)"* should create a stack that can contain at most 10 elements. Now, create a manual exception named *StackException* that will accept some message *msg,* and on print will output "*StackException: "* followed by *msg.* Popping an empty stack should throw the exception with message "Stack underflow", and pushing to a filled up stack must throw the same with message "Stack overflow". Next create two threads, one will push from 1 to 15, and another thread that will pop 15 times. With suitable sleep time of the threads, simulate and show that unsynchronized stack may throw both exceptions.

3. Now, synchronize the push and pop operations of the Stack. Will it solve the problem?

4. Implement the push and pop operations in such a way, such that the push will wait if the stack is full, and the pop will wait if the stack is empty. You have created a thread-safe Stack.

5. Implement the two threads, one performing the stated push and the other performing pop as in Problem 2, as anonymous inner classes within the main method itself.