**OOP LAB**

**Session III**

**Part I**

**Lab No. 7:**  Interfaces and Exception Handling

Lab Exercises

1. Design a stack class. Provide your own stack exceptions namely Push Exception and Pop Exception, which throw exceptions when the stack is full and when the stack is empty respectively. Show the usage of these exceptions in handling a stack object in the main.
2. Define a class CurrentDate with data members day, month and year. Define a method createDate() to create date object by reading values from keyboard. Throw a user defined exception by name InvalidDayException if the day is invalid and InvalidMonthException if month is found invalid and display current date if the date is valid. Write a test program to illustrate the functionality.
3. Design a Student class with appropriate data members as in Lab 5. Provide your own exceptions namely Seats Filled Exception, which is thrown when Student registration number is >XX25 (where XX is last two digits of the year of joining) Show the usage of this exception handling using Student objects in the main. (Note: Registration number must be a unique number)

**Part II**

**Lab No. 8:** Multithreading

Lab Exercises

1. Create a class by extending Thread Class to print a multiplication table of a number supplied as parameter. Create another class Tables which will instantiate two objects of the above class to print multiplication table of 5 and 7.
2. Write and execute a java program to create and initialize a matrix of integers. Create n threads( by implementing Runnable interface) where n is equal to the number of rows in the matrix. Each of these threads should compute a distinct row sum. The main thread computes the complete sum by looking into the partial sums given by the threads.
3. Write and execute a java program to implement a producer and consumer problem using Inter-thread communication.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\***