OOP LAB

Week 6 Assignment Submission

Swamiraju Satya Praveen Varma

200905044

Batch B1

10

1) Create a class MxNTableThread by extending Thread class. The thread calls a non- static printTable method of another class to display multiplication table of a number supplied as parameter. Create another class TablesDemo which will instantiate two objects of the MxNTableThread class to print multiplication table of 5 and 7. Observe intermixed output from the 2 threads. Also, observe output by applying synchronization concept.

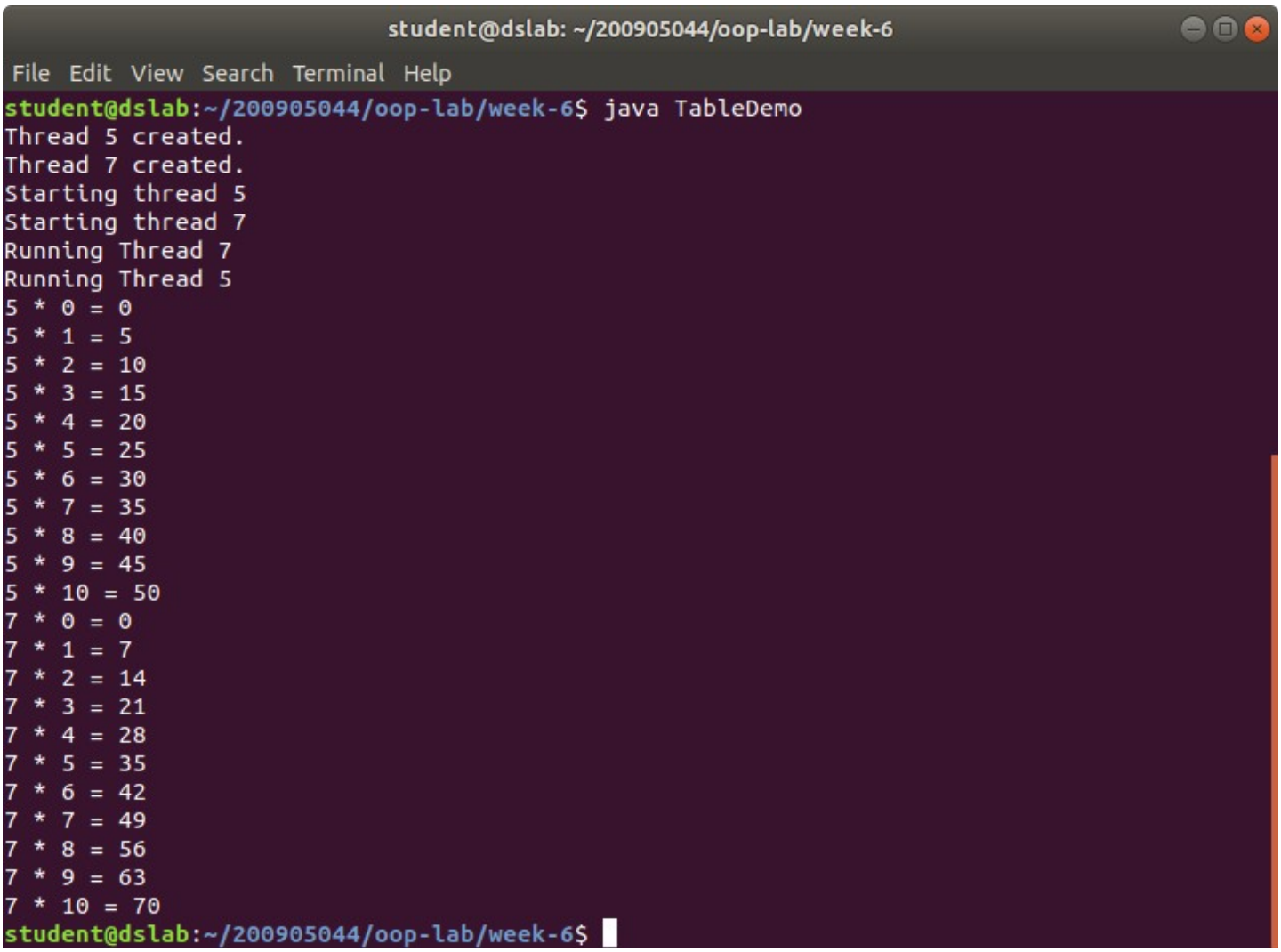
Code:

```
class MultiTables extends Thread
{
	private int num;
	Thread t;

	public MultiTables(int num)
	{
		this.num = num;
		System.out.println("Thread " + this.num + " created.");
	}

	public void printMultiTables()
	{
```

```
        for(int i = 0; i<11; i++)
        {
            System.out.println(this.num + " * " + i + " = " + +(i*this.num));
        }
    }

    public void run()
    {
        System.out.println("Running Thread " + this.num);
        this.printMultiTables();
    }

    public void startThrd()
    {
        System.out.print("Starting thread " + this.num + "\n");
        if(t == null)
        {
            t = new Thread(this, "Thread " + this.num);
            t.start();
        }
    }
}

public class TableDemo
{
    public static void main(String[] args) {
        MultiTables t1 = new MultiTables(5);
```

```
        MultiTables t2 = new MultiTables(7);
        t1.startThrd();
        try
        {
            t1.join();
        }
        catch (InterruptedException e)
        {
            e.printStackTrace();
        }
         t2.startThrd();
    }
}
```

Sample input/output:



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. Use join method to ensure that the main thread terminates last.

Code:

```
import java.util.Scanner;


class Matrix
{
```

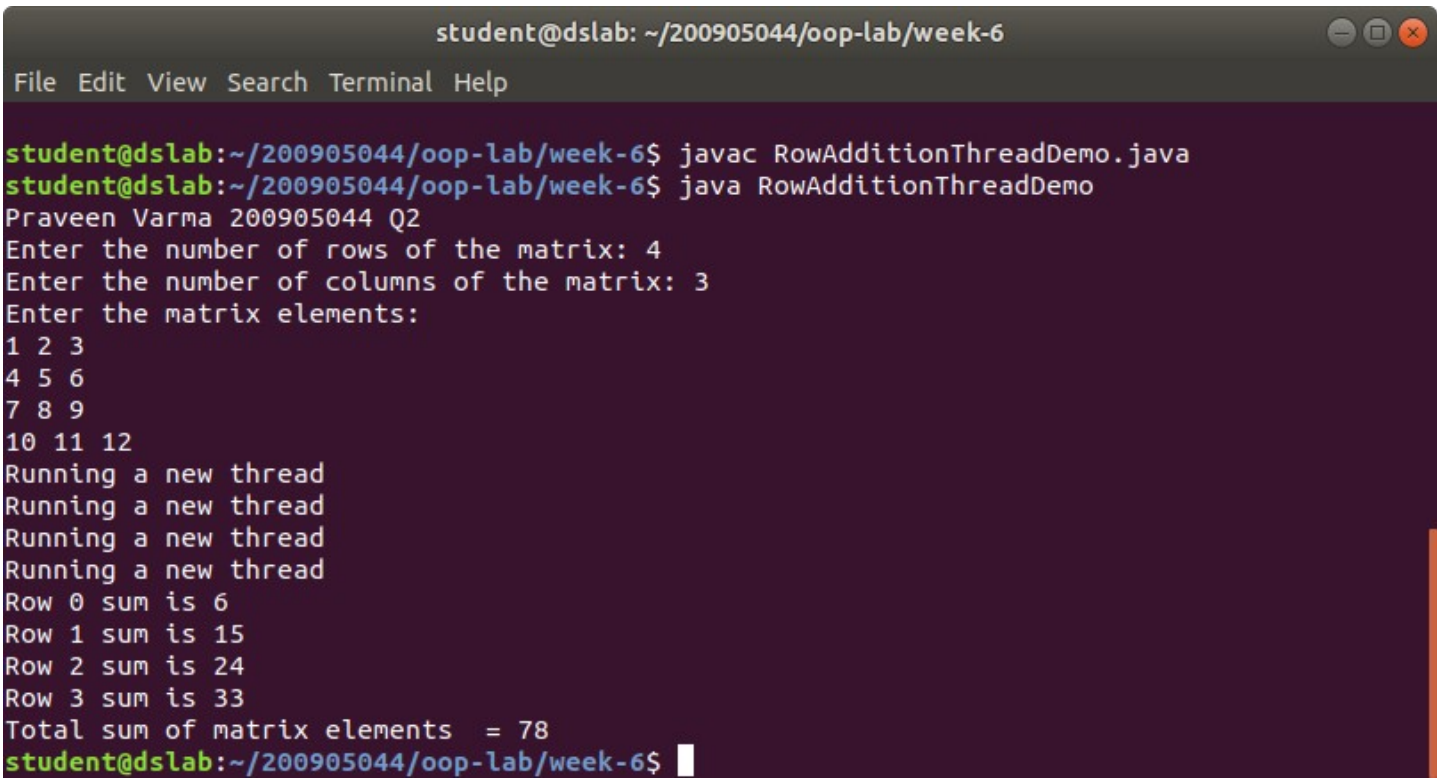
```
    int arr[][];

    Matrix(int n, int m)
    {
        arr=new int[n][m];
    }

    int[] getRow (int i)
    {
        return arr[i];
    }

    void initialize ()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the matrix elements:");
        for (int i = 0;i<arr.length;i++)
        {
            for (int j = 0; j < arr[i].length; j++)
            {
                arr[i][j] = sc.nextInt();
            }
        }
    }

}
```

```
class RowAddition implements Runnable
{

    int arr[];
    int sum;
    int row_no;
    RowAddition (int a[],int rno)
    {
        arr = a;
        sum = 0;
        row_no=rno;
    }

    public int getRowAddition ()
    {
        return sum;
    }

    public void run ()

    {
        System.out.println("Running a new thread");
        for (int i=0;i<arr.length;i++)
            sum += arr[i];
        System.out.println("Row "+row_no+" sum is "+sum);
    }
```

```
}

public class RowAdditionThreadDemo
{

    public static void main (String [] args)
    {

        Scanner sc = new Scanner(System.in);
    System.out.println("Praveen Varma 200905044 Q2");
        System.out.print("Enter the number of rows of the matrix: ");
        int r = sc.nextInt();
        System.out.print("Enter the number of columns of the matrix: ");
        int c = sc.nextInt();

        Matrix matrix = new Matrix(r, c);
        matrix.initialize();

        Thread threads[] = new Thread[r];
        RowAddition RowAddition[] = new RowAddition[r];

        for (int i = 0; i < r; ++i)
        {
            RowAddition[i] = new RowAddition(matrix.getRow(i),i);
            threads[i] = new Thread(RowAddition[i]);
            threads[i].start();
        }
```

```
        int sum = 0;

        try
        {
            for (int i = 0; i < r; ++i)
            {
                threads[i].join();
                sum += RowAddition[i].getRowAddition();
            }
        }
        catch (InterruptedException e)
        {
            System.out.println("Thread interupted");
        }

        System.out.println("Total sum of matrix elements  = " + sum);

    }
}
```

Sample input/output:



3) Write and execute a java program to implement a producer - consumer problem using Inter-thread communication.

Code:

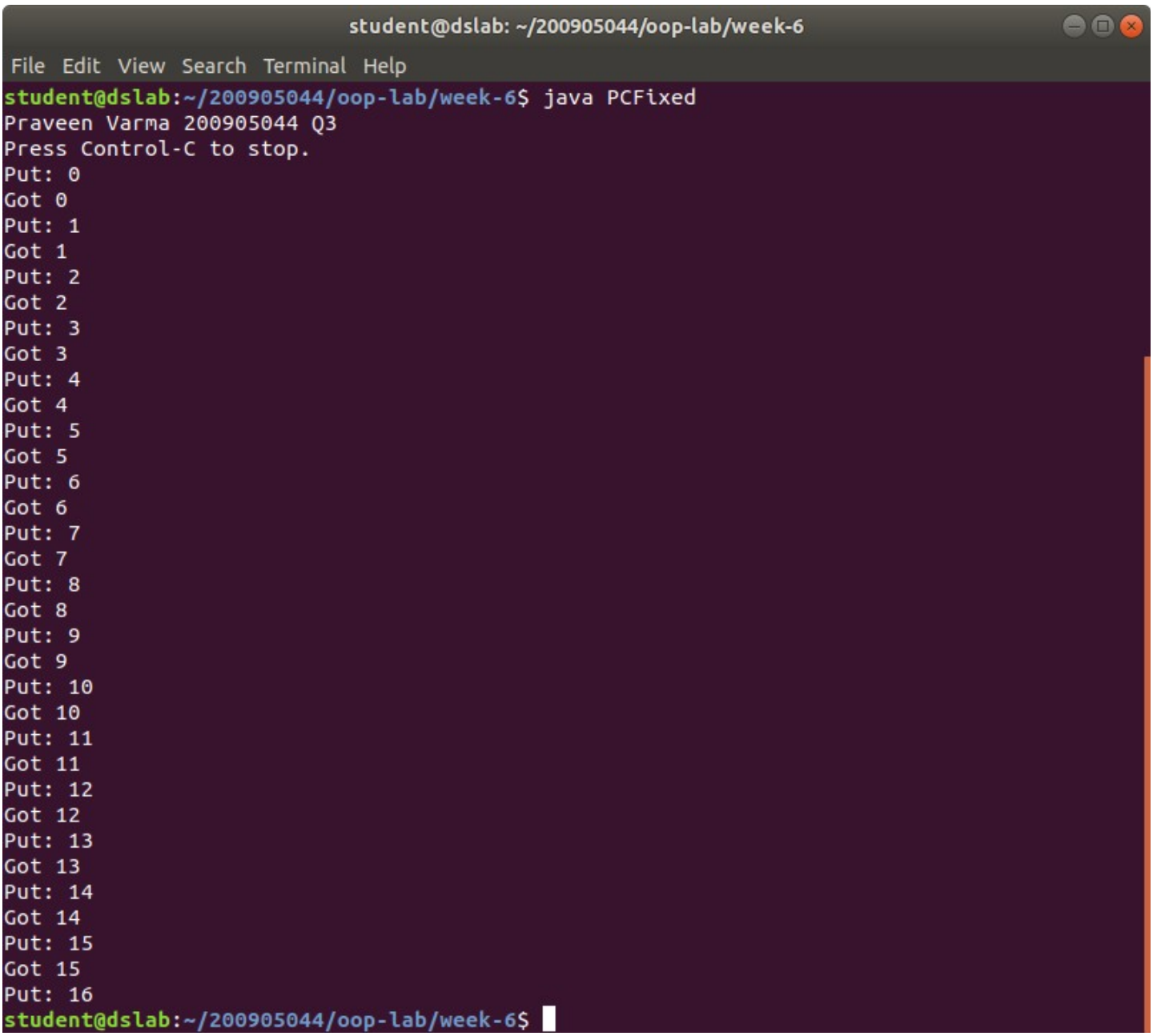
```
class Q
{
 int n;
 boolean valueSet = false;
 synchronized int get()
 {
            while(!valueSet)
               try
               {
                  wait();
                  Thread.sleep(200);
```

```
            }
            catch(InterruptedException e)
            {
                System.out.println("InterruptedException caught");
            }
        System.out.println("Got " + n);
        valueSet = false;
        notify();
        return n;
    }
    synchronized void put(int n)
    {
            while(valueSet)
                    try
                    {
                            wait();
                    }
            catch(InterruptedException e)
            {
                System.out.println("InterruptedException caught");
            }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        notify();
    }
}
```

```
class Producer implements Runnable
{
  Q q;
  Producer(Q q)
  {
     this.q = q;
        new Thread(this, "Producer").start();
  }
  public void run()
  {
     int i = 0;
     while(true)
     {
        q.put(i++);
     }
  }
}

class Consumer implements Runnable
{
  Q q;
  Consumer(Q q)
  {
    this.q = q;
    new Thread(this, "Consumer").start();
  }
  public void run()
```

```
  {
    while(true)
    {
                    q.get();
                }
  }
}
public class PCFixed
{
   public static void main(String[] args)
   {
      System.out.println("Praveen Varma 200905044 Q3");
                Q q = new Q();
                new Producer(q);
                new Consumer(q);
                System.out.println("Press Control-C to stop.");
   }

}
```

Sample input/output:


student@dslab: ~/200905044/oop-lab/week-6
File Edit View Search Terminal Help
student@dslab:~/200905044/oop-lab/week-6$ java PCFixed
Praveen Varma 200905044 Q3
Press Control-C to stop.
Put: 0
Got 0
Put: 1
Got 1
Put: 2
Got 2
Put: 3
Got 3
Put: 4
Got 4
Put: 5
Got 5
Put: 6
Got 6
Put: 7
Got 7
Put: 8
Got 8
Put: 9
Got 9
Put: 10
Got 10
Put: 11
Got 11
Put: 12
Got 12
Put: 13
Got 13
Put: 14
Got 14
Put: 15
Got 15
Put: 16
student@dslab:~/200905044/oop-lab/week-6$

THANK YOU!