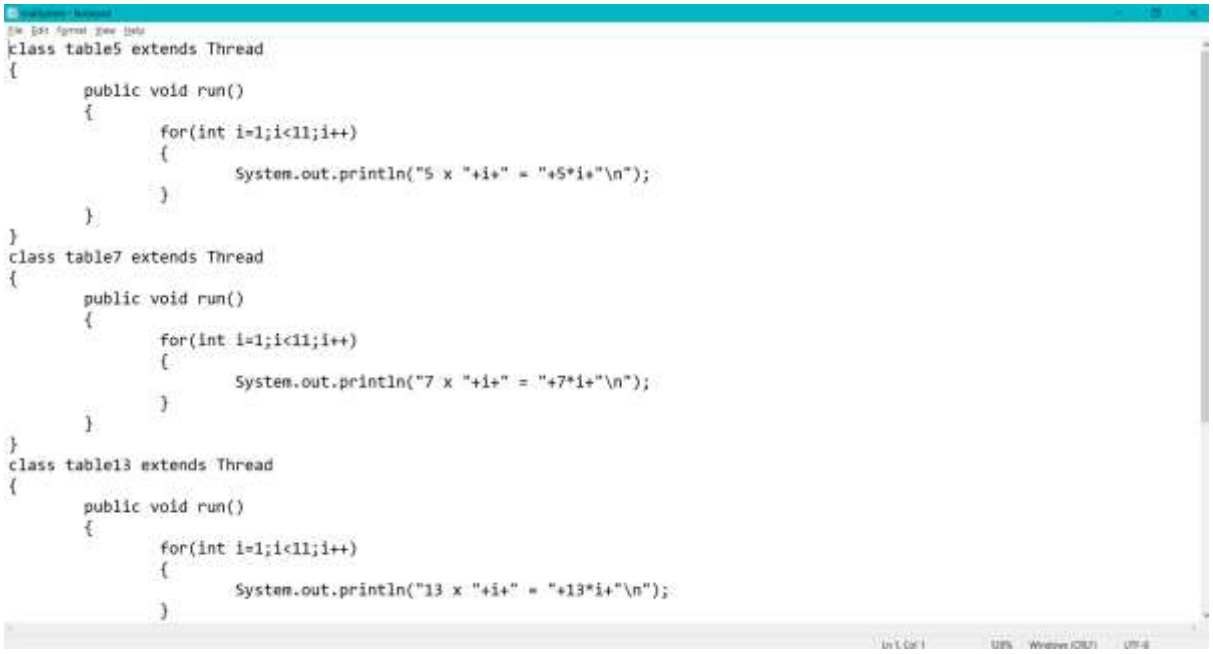
LAB 10: MULTITHREADING IN JAVA

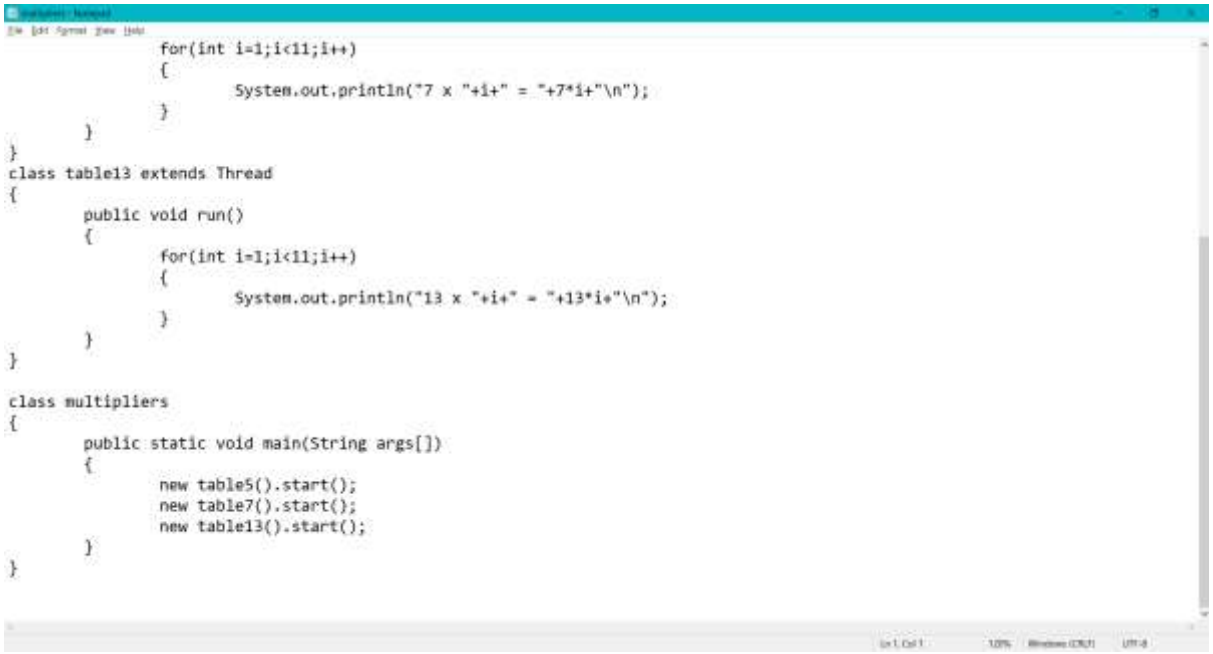
Name: Shreyas Sawant Div: D7A Roll No.: 55

Q.1 Write a program to print the table of 5,7,13 using multithreading (use thread class)

CODE:

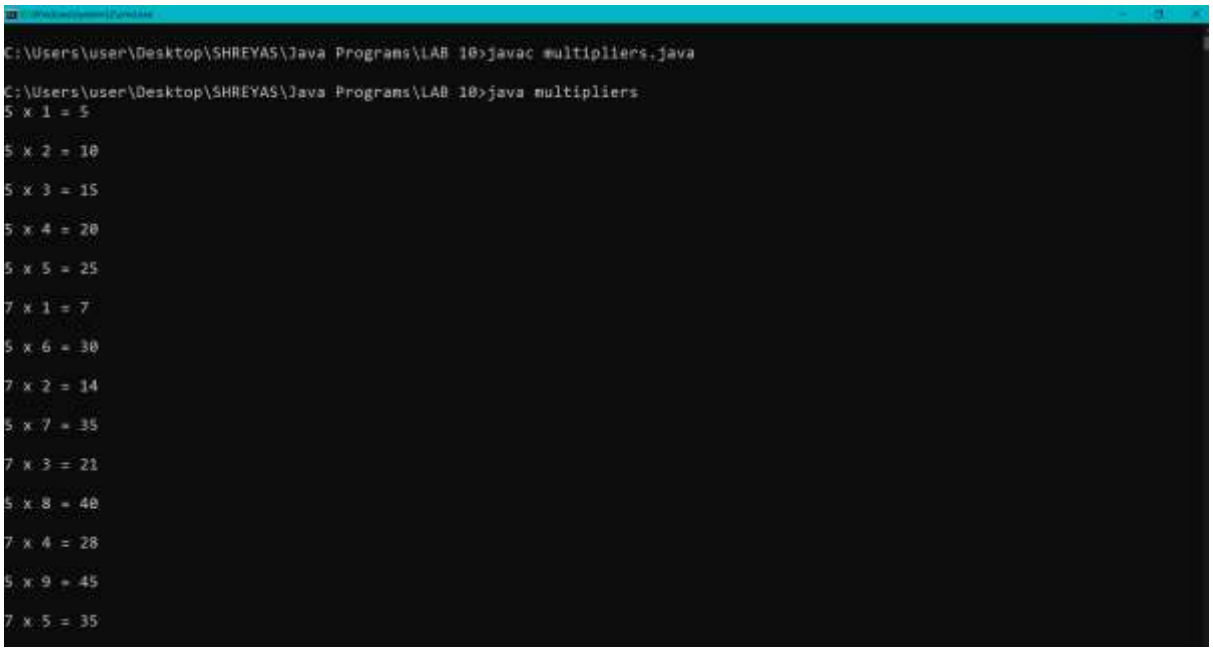


```
class table5 extends Thread
{
	public void run()
	{
		for(int i=1;i<11;i++)
		{
			System.out.println("5 x "+i+" = "+5*i+"\n");
		}
	}
}
class table7 extends Thread
{
	public void run()
	{
		for(int i=1;i<11;i++)
		{
			System.out.println("7 x "+i+" = "+7*i+"\n");
		}
	}
}
class table13 extends Thread
{
	public void run()
	{
		for(int i=1;i<11;i++)
		{
			System.out.println("13 x "+i+" = "+13*i+"\n");
		}
```



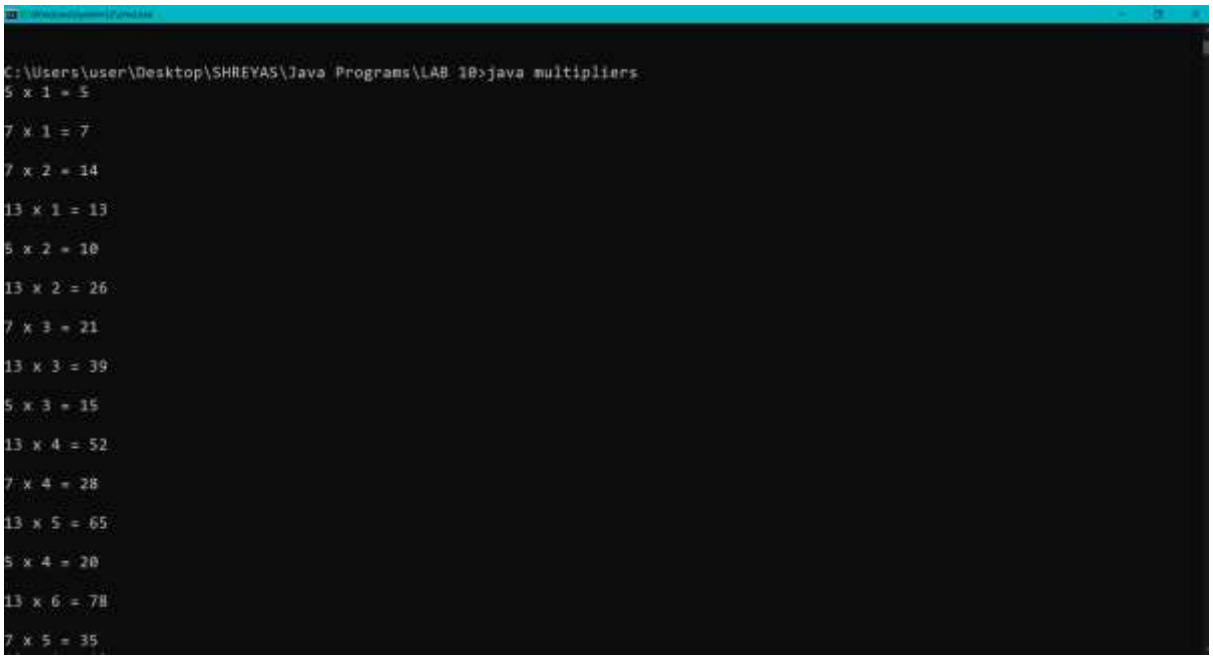
```
		for(int i=1;i<11;i++)
		{
			System.out.println("7 x "+i+" = "+7*i+"\n");
		}
	}
}
class table13 extends Thread
{
	public void run()
	{
		for(int i=1;i<11;i++)
		{
			System.out.println("13 x "+i+" = "+13*i+"\n");
		}
	}
}

class multipliers
{
	public static void main(String args[])
	{
		new table5().start();
		new table7().start();
		new table13().start();
	}
}
```

OUTPUT:


C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>javac multipliers.java
C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>java multipliers
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
7 x 1 = 7
5 x 6 = 30
7 x 2 = 14
5 x 7 = 35
7 x 3 = 21
5 x 8 = 40
7 x 4 = 28
5 x 9 = 45
7 x 5 = 35


7 x 5 = 35
13 x 1 = 13
5 x 10 = 50
13 x 2 = 26
7 x 6 = 42
13 x 3 = 39
7 x 7 = 49
13 x 4 = 52
7 x 8 = 56
13 x 5 = 65
7 x 9 = 63
13 x 6 = 78
7 x 10 = 70
13 x 7 = 91
13 x 8 = 104
13 x 9 = 117


5 x 10 = 50
13 x 2 = 26
7 x 6 = 42
13 x 3 = 39
7 x 7 = 49
13 x 4 = 52
7 x 8 = 56
13 x 5 = 65
7 x 9 = 63
13 x 6 = 78
7 x 10 = 70
13 x 7 = 91
13 x 8 = 104
13 x 9 = 117
13 x 10 = 130


C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>java multipliers
5 x 1 = 5
7 x 1 = 7
7 x 2 = 14
13 x 1 = 13
5 x 2 = 10
13 x 2 = 26
7 x 3 = 21
13 x 3 = 39
5 x 3 = 15
13 x 4 = 52
7 x 4 = 28
13 x 5 = 65
5 x 4 = 20
13 x 6 = 78
7 x 5 = 35


13 x 7 = 91
5 x 5 = 25
13 x 8 = 104
7 x 6 = 42
13 x 9 = 117
5 x 6 = 30
13 x 10 = 130
7 x 7 = 49
5 x 7 = 35
7 x 8 = 56
5 x 8 = 40
7 x 9 = 63
5 x 9 = 45
7 x 10 = 70
5 x 10 = 50
C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>

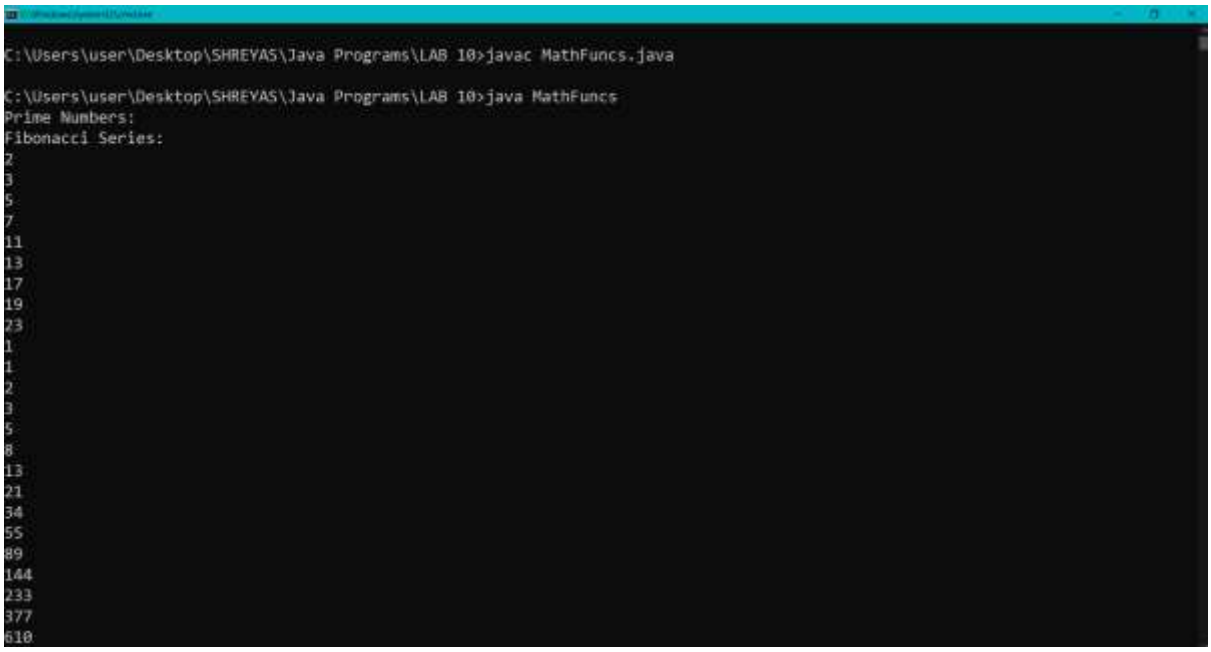
Q.2 Write a program to print first 20 prime numbers and 15 fibonacci terms using runnable interface.

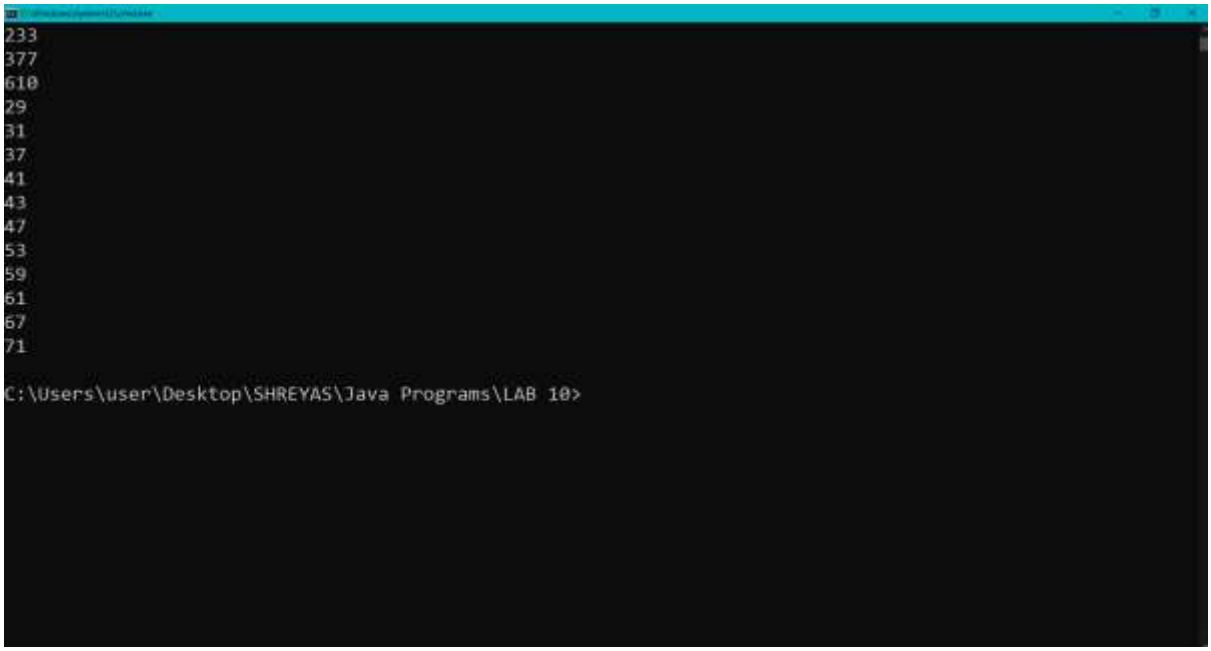
CODE:

```
class Prime implements Runnable
{
	public void run()
	{	System.out.println("Prime Numbers: ");
		int k=0;int i=2;
		while(k!=20)
		{
			if(prime(i))
			{ System.out.println(i);
				k++;
			}
			i++;
		}
	}

	boolean prime(int a)
	{	int k=0;
		for(int i=1;i<a;i++)
		{	if(a%i==0)
				k++;
		}
		if(k==1)
			return true;
		else
			return false;
	}
}
class Fibs implements Runnable
```

```
class Fibs implements Runnable
{
	public void run()
	{
		System.out.println("Fibonacci Series: ");
		int a=0,b=1,c,k=0;
		System.out.println(a+b);
		while(k!=14)
		{
			c=a+b;
			a=b;
			b=c;
			System.out.println(c);
			k++;
		}
	}
}
class MathFuncs
{
	public static void main(String args[])
	{
		Prime runnable=new Prime();
		Fibs runnable1=new Fibs();
		new Thread(new Prime()).start();
		new Thread(new Fibs()).start();
	}
}
```

OUTPUT:


C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>javac MathFuncs.java
C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>java MathFuncs
Prime Numbers:
Fibonacci Series:
2
3
5
7
11
13
17
19
23
1
1
2
3
5
8
13
21
34
55
89
144
233
377
610


233
377
610
29
31
37
41
43
47
53
59
61
67
71
C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>

Q.3. Write a program to demonstrate concept of synchronisation

CODE:

Non-synchronised method

```
class Table
{
        void printTable(int n)
        {       System.out.println("Table of "+n+": ");
                for(int i=1;i<=5;i++)
                        System.out.println(n*i);
        }
}
public class NonSynchro
{
        public static void main(String args[])
        {
                Table obj = new Table();
                Thread t1=new Thread(){
                public void run(){
                        obj.printTable(5);
                }
                };
                Thread t2=new Thread(){
                public void run(){
                        obj.printTable(13);
                }
                };
                t1.start();
                t2.start();
        }
}
```

OUTPUT:

```
C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>java NonSynchro
Table of 5:
Table of 13:
13
26
39
52
65
5
10
15
20
25

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>java NonSynchro
Table of 13:
Table of 5:
5
10
15
20
25
13
26
39
52
65

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>
```

CODE:

Synchronised method

```
class Table
{
        synchronized void printTable(int n)
        {       System.out.println("\nTable of "+n+": ");
                for(int i=1;i<=10;i++)
                        System.out.println(n*i);
        }
}

public class Synchro
{
        public static void main(String args[])
        {
                Table obj = new Table();
                Thread t1=new Thread(){
                public void run(){
                        obj.printTable(5);
                }
                };
                Thread t2=new Thread(){
                public void run(){
                        obj.printTable(13);
                }
                };
                t1.start();
                t2.start();
        }
}
```

OUTPUT:

```
C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>javac Synchro.java

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>java Synchro

Table of 5:
5
10
15
20
25
30
35
40
45
50

Table of 13:
13
26
39
52
65
78
91
104
117
130

C:\Users\user\Desktop\SHREYAS\Java Programs\LAB 10>
```