Name: Anish Ashok Sharma **Sap id: 60003220045**

Branch: Information Technology **Div: D/IT1**

Course: **Object Oriented Programming using Java**

Experiment no. 11

Aim: To implement Multithreading

Problem Statement 1:
Write a multithreaded program a java program to print Table of Five, Seven and Thirteen using Multithreading (Use Thread class for the implementation)

Code:

```
class Table1 extends Thread
{
	public void run()
	{
		for(int i=1;i<=10;i++)
		{
			System.out.println(5+"*"+i+"="+5*i);
		}
	}
}
class Table2 extends Thread
{
	public void run()
	{
		for(int i=1;i<=10;i++)
		{
			System.out.println(7+"*"+i+"="+7*i);
		}
	}
}
class Table3 extends Thread
{
	public void run()
	{
		for(int i=1;i<=10;i++)
		{
			System.out.println(13+"*"+i+"="+13*i);
		}
	}
}
class Thread1
{
	public static void main(String[] args) throws Exception
	{
		Table1 obj1=new Table1();
		obj1.start();
		Table2 obj2=new Table2();
		obj2.start();
		Table3 obj3=new Table3();
```

```
            obj3.start();
        }
}
```

Output

```
5*1=5
5*2=10
7*1=7
13*1=13
5*3=15
7*2=14
13*2=26
13*3=39
5*4=20
7*3=21
7*4=28
7*5=35
13*4=52
5*5=25
7*6=42
13*5=65
5*6=30
7*7=49
13*6=78
13*7=91
13*8=104
5*7=35
5*8=40
5*9=45
5*10=50
7*8=56
13*9=117
7*9=63
13*10=130
7*10=70
```

Problem Statement 2:

Write a multithreaded program to display /*/*/*/*/*/*/*/* using 2 child threads.
Code:

```
class A extends Thread
{
	public void run()
	{



			try{
				for(int i=1;i<=8;i++)
				{
					System.out.print("/");
					Thread.sleep(498);
				}
			}
			catch(Exception e){
				System.out.println(e);
			}




	}
}
class B extends Thread
{
	public void run()
	{
		try{
			for(int i=1;i<=8;i++)
			{
				System.out.print("*");
				Thread.sleep(498);
```

```
                }
            }
            catch(Exception e){
                System.out.println(e);
            }

        }
}
class ThreadPattern
{
    public static void main(String[] args)
    {
        A obj1=new A();
        obj1.start();
        B obj2=new B();
        obj2.start();
    }
}
```

Output

```
C:\Users\91720\OneDrive\Desktop\Anish Java>java ThreadPattern
/*/*/*/*/*/*/*/*
```

Problem Statement 3:

Write a multithreaded program that generates the Fibonacci sequence. This program should work as follows: create a class Input that reads the number of Fibonacci numbers that the program is to generate. The class will then create a separate thread that will generate the Fibonacci numbers, placing the sequence in an array. When the thread finishes execution, the parent thread (Input class) will output the sequence generated by the child thread. Because the parent thread cannot begin outputting the Fibonacci sequence until the child thread finishes, the parent thread will have to wait for the child thread to finish.

Code:

```
import java.util.*;

class input

{

	static int n,a[];

	public static void main(String ar[])

	{

		Scanner sc=new Scanner(System.in);

		System.out.println("enter the number");

		n=sc.nextInt();

		a=new int[n];

		fibo f=new fibo();

		Thread t=new Thread(f);

		t.start();

		int ctr=0;

		while(t.isAlive())

		{

			try

			{

				System.out.print(ctr++);

				Thread.sleep(1);

			}

			catch(Exception e)

			{

			}
```

```
                }
                for(int i=1;i<n;i++)
                {
                        System.out.print(" "+a[i]);
                }
        }
}
class fibo extends input implements Runnable
{
        public void run()
        {
                a[0]=0;
                a[1]=1;
                for(int i=2;i<n;i++)
                {
                        a[i]=a[i-1]+a[i-2];
                }
        }
}
```

Output

```
C:\Users\91720\OneDrive\Desktop\Anish Java>java input
enter the number
10
0 1 1 2 3 5 8 13 21 34
```

Problem Statement 4:

WAP to prevent concurrent booking of a ticket using the concept of thread synchronization.

Code:

```
import java.util.*;
class TB
{
	Scanner sc=new Scanner(System.in);
	synchronized public void bookTicket()
	{
		int fare;
		try
		{
			System.out.println("enter the fare of the ticket");
			fare=sc.nextInt();
			System.out.print("Processing...");
			for(int i=0;i<8;i++)
			{
				System.out.print(".. ");
				Thread.sleep(500);
			}
			System.out.println();
		}
		catch(Exception e)
		{
			System.out.println(e);
		}
		System.out.println("booking successfully confirmed");
	}
}
class Passenger extends Thread
{
```

```
        TB ticket;
        Passenger(TB ticket)
        {
                this.ticket=ticket;
        }
        public void run()
        {
                ticket.bookTicket();
        }
}
class BT
{
        public static void main(String ar[])
        {
                TB obj=new TB();
                Passenger p1=new Passenger(obj);
                Passenger p2=new Passenger(obj);
                p1.start();
                p2.start();
        }
}
```

Output

```
C:\Users\91720\OneDrive\Desktop\Anish Java>java BT
enter the fare of the ticket
400
Processing..... .. .. .. .. .. .. ..
booking successfully confirmed
enter the fare of the ticket
300
Processing..... .. .. .. .. .. .. ..
booking successfully confirmed
```