WEEK- 9

JAVA ASSIGNMENT

Code 1:

Single Threading Using Thread:

```
class Add extends Thread{
   int num1,num2;
   Add(int a, int b){
      num1=a; num2=b;
   }
   public void run(){
   for(int i=1; i<5; i++)
      System.out.println("sum=" + (num1+i) + "+" + (num2+i) +"="+ ((num1+i)
+ (num2+i)));
   }
}
class Singlethreading1{
public static void main(String[] args){
       System.out.println("Name:Satlas Rohit B\nRegno:2024503305");
      Add aobj=new Add(5,3);
      aobj.start();
   }
}
```

Output:

```
Name:Satlas Rohit B
Regno:2024503305
sum=6+4=10
sum=7+5=12
sum=8+6=14
sum=9+7=16
```

MultiThreading using Thread:

```
class Main{
public static void main(String[] args){
        System.out.println("Name:Satlas Rohit B\nRegno:2024503305");
        Add aobj=new Add(5,3);
        Sub sobj=new Sub(5,3);
        aobj.start();
        sobj.start();
        try{
           aobj.join();
           sobj.join();
        }
        catch(Exception e){
           System.out.println(e);
        }
    }
}
class Add extends Thread{
    int num1,num2;
    Add(int a, int b){
        num1=a; num2=b;
    }
    public void run(){
    for(int i=1; i<5; i++)
        System.out.println("sum=" + (num1+i) + "+" + (num2+i) +"="+ ((num1+i)
+ (num2+i)));
```

```
    }
}
class Sub extends Thread{
    int num1,num2;
    Sub(int a, int b){
        num1=a; num2=b;
    }
    public void run(){
        for(int i=1; i<5; i++)
            System.out.println("difference=" + (num1+i) + "-" + (num2+i)+"="+
((num1+i) - (num2+i)));
        }
}
```

Output:

```
Name:Satlas Rohit B
Regno:2024503305
difference=6-4=2
difference=7-5=2
difference=8-6=2
difference=9-7=2
sum=6+4=10
sum=7+5=12
sum=8+6=14
sum=9+7=16
```

Code 2:

Single Threading Using Runnable:

```
class Main{
public static void main(String[] args){
        System.out.println("Name:Satlas Rohit B\nRegno:2024503305");
```

```
        Add aobj=new Add(5,3);
        Thread t1=new Thread(aobj);
        t1.start();
    }
}

class Add implements Runnable{
    int num1,num2;
    Add(int a, int b){
        num1=a; num2=b;
    }
    public void run(){
        for(int i=1; i<5; i++)
            System.out.println("sum=" + (num1+i) + "+" + (num2+i) +"="+
((num1+i) + (num2+i)));
    }

}
```

Output:

```
Name:Satlas Rohit B
Regno:2024503305
sum=6+4=10
sum=7+5=12
sum=8+6=14
sum=9+7=16
```

Multithreading using Runnable:

```
class Main{
public static void main(String[] args){
      System.out.println("Name:Satlas Rohit B\nRegno:2024503305");
      Add aobj=new Add(5,3);
      Sub sobj=new Sub(5,3);
      Thread t1=new Thread(aobj);
      Thread t2=new Thread(sobj);
      t1.start();
      t2.start();
   }
}
class Add implements Runnable{
   int num1,num2;
   Add(int a, int b){
      num1=a; num2=b;
   }
   public void run(){
   for(int i=1; i<5; i++)
      System.out.println("sum=" + (num1+i) + "+" + (num2+i) +"="+ ((num1+i)
+ (num2+i)));
   }
}
class Sub implements Runnable{
   int num1,num2;
   Sub(int a, int b){
```

```
        num1=a; num2=b;
    }
    public void run(){
        for(int i=1; i<5; i++)
            System.out.println("difference=" + (num1+i) + "-" + (num2+i)+"="+
((num1+i) - (num2+i)));
        }
}
```

Output:

```
Name:Satlas Rohit B
Regno:2024503305
difference=6-4=2
difference=7-5=2
difference=8-6=2
difference=9-7=2
sum=6+4=10
sum=7+5=12
sum=8+6=14
sum=9+7=16
```

Code 3:

Lambda

```
class lambda implements Runnable{
    public static void main(String[] args){
        System.out.println("Name:Satlas Rohit B\nRegno:2024503305");
        int num1=5,num2=3;
        Thread t1=new Thread(()->{
            for(int i=1; i<5; i++)
                System.out.println("sum=" + (num1+i) + "+" + (num2+i) +"="+
((num1+i) + (num2+i)));
```

```
        });
        t1.start();
    }
    public void run(){
    }
}
```

Output:

```
Name:Satlas Rohit B
Regno:2024503305
sum=6+4=10
sum=7+5=12
sum=8+6=14
sum=9+7=16
```

Code 4:

Synchronization

```
public class Synchronization{
    public static void main(String[] args) {
        System.out.println("Name:Satlas Rohit B\nRegno:2024503305");
        Thread t1=new Thread(()->disp("Java"));
        Thread t2=new Thread(()->disp("Python"));
        t1.start();
        t2.start();
    }
    public static synchronized void disp(String lang){
            for(int i=0;i<lang.length();i++){
                System.out.print(lang.charAt(i));
                try{
```

```
            Thread.sleep(1000);
          }
          catch(Exception e){
            e.printStackTrace();
          }
        }
      }
}
```

Output:

```
Name:Satlas Rohit B
Regno:2024503305
JavaPython
```

Code 5:

Semaphores

```
import java.util.concurrent.Semaphore;
public class Semaphores {
   public static void main(String[] args) {
      System.out.println("Name:Satlas Rohit B\nRegno:2024503305");
      ParkingLot lot = new ParkingLot(3);
      for (int i = 1; i <= 6; i++) {
         final String carName = "Car-" + i;
         new Thread(() -> lot.parkCar(carName)).start();
      }
   }
}
```

```
class ParkingLot {
    private final Semaphore parkingSlots;
    public ParkingLot(int totalSpots) {
        parkingSlots = new Semaphore(totalSpots);
    }
    public void parkCar(String carName) {
        try {
            System.out.println(carName + " is trying to park...");
            parkingSlots.acquire();
            System.out.println(carName + " has parked.");
            Thread.sleep(2000);
            System.out.println(carName + " is leaving the parking lot...");
        } catch (InterruptedException e) {
            e.printStackTrace();
        } finally {
            parkingSlots.release();
            System.out.println(carName + " left. Available spots: " +
parkingSlots.availablePermits());
        }
    }
}
```

Output:

```
Name:Satlas Rohit B
Regno:2024503305
Car-2 is trying to park...
Car-2 has parked.
Car-4 is trying to park...
Car-4 has parked.
Car-1 is trying to park...
Car-1 has parked.
Car-3 is trying to park...
Car-6 is trying to park...
Car-5 is trying to park...
Car-2 is leaving the parking lot...
Car-1 is leaving the parking lot...
Car-4 is leaving the parking lot...
Car-5 has parked.
Car-6 has parked.
Car-3 has parked.
Car-1 left. Available spots: 1
Car-4 left. Available spots: 1
Car-2 left. Available spots: 1
Car-5 is leaving the parking lot...
Car-3 is leaving the parking lot...
Car-3 left. Available spots: 2
Car-6 is leaving the parking lot...
Car-5 left. Available spots: 1
Car-6 left. Available spots: 3
```

Code 6:

Deadlock

```
public class DeadlockDemo {
    public static void main(String[] args) {
        System.out.println("Name:Satlas Rohit B\nRegno:2024503305");
        final Resource resource1 = new Resource("Resource 1");
        final Resource resource2 = new Resource("Resource 2");
```

```
// Thread 1 tries to lock resource1, then resource2
Thread t1 = new Thread(() -> {
    synchronized (resource1) {
        System.out.println("Thread 1: locked " + resource1.name);
        try { Thread.sleep(100); } catch (InterruptedException e) {}
        System.out.println("Thread 1: waiting to lock " + resource2.name);
        synchronized (resource2) {
            System.out.println("Thread 1: locked both resources");
        }
    }
});

// Thread 2 tries to lock resource2, then resource1
Thread t2 = new Thread(() -> {
    synchronized (resource2) {
        System.out.println("Thread 2: locked " + resource2.name);
        try { Thread.sleep(100); } catch (InterruptedException e) {}
        System.out.println("Thread 2: waiting to lock " + resource1.name);
        synchronized (resource1) {
            System.out.println("Thread 2: locked both resources");
        }
    }
});

t1.start();
```

```
        t2.start();
    }
}
class Resource {
    String name;
    Resource(String name) {
        this.name = name;
    }
}
```

Output:

```
Name:Satlas Rohit B
Regno:2024503305
Thread 2: locked Resource 2
Thread 1: locked Resource 1
Thread 2: waiting to lock Resource 1
Thread 1: waiting to lock Resource 2
```