Week 06 : Programming Assignment 4

Due on 2025-03-06, 23:59 IST

Execution of two or more threads occurs in a random order. The keyword `synchronized` in Java is used to control the execution of threads in a specific sequence. In the following program, some numbers are expected to be printed. Use the `synchronized` keyword appropriately to ensure that the program prints the output in the following order:

```
----------OUTPUT----------
5
10
15
20
25
100
200
300
400
500
--------------------------------------------------
```

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1		5\n 10\n 15\n 20\n 25\n 100\n 200\n 300\n 400\n 500	5\n 10\n 15\n 20\n 25\n 100\n 200\n 300\n 400\n 500\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-03-05, 19:46 IST

Your last recorded submission was :

```java
class Execute{
    synchronized void print(int n) {
        for (int i = 1; i <= 5; i++) {
            System.out.println(n * i);
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                System.out.print(e);
            }
        }
    }

} // Ending Execute class

class Thread1 extends Thread{
    Execute t;
    Thread1(Execute t){
        this.t=t;
    }
    public void run(){
        t.print(5);
    }
}

class Thread2 extends Thread{
    Execute t;
    Thread2(Execute t){
        this.t=t;
    }
    public void run(){
        t.print(100);
    }
}

public class W06_P4{
    public static void main(String args[]){
        Execute obj = new Execute();//only one object
        Thread1 t1=new Thread1(obj);
        Thread2 t2=new Thread2(obj);
        t1.start();
        t2.start();
    }
}
```

Sample solutions (Provided by instructor)

```java
class Execute{
// Just add 'synchronized' in the method
synchronized void print(int n){
   for(int i=1;i<=5;i++){
     System.out.println(n*i);
     try{
      Thread.sleep(400);
     }catch(Exception e){
        System.out.println(e);
     }
   }
 }
} // Ending Execute class

class Thread1 extends Thread{
    Execute t;
    Thread1(Execute t){
        this.t=t;
    }
    public void run(){
        t.print(5);
    }
}

class Thread2 extends Thread{
    Execute t;
    Thread2(Execute t){
        this.t=t;
    }
    public void run(){
        t.print(100);
    }
}

public class W06_P4{
    public static void main(String args[]){
        Execute obj = new Execute();//only one object
        Thread1 t1=new Thread1(obj);
        Thread2 t2=new Thread2(obj);
        t1.start();
        t2.start();
    }
}
```