

## Java Week 6: Q4

Due on 2020-10-29, 23:59 IST

## Course outline

How does an NPTEL online course work?

Week 0 : Assignment 0

Week 1 :

Week 2 :

Week 3 :

Week 4 :

Week 5 :

Week 6 :

- Lecture 26 : Demonstration-X
- Lecture 27 : Multithreading-I
- Lecture 28 : Multithreading-II
- Lecture 29 : Demonstration-XI
- Lecture 30 : I-O Stream-I
- Quiz: Assignment 6
- Java Week 6: Q1
- Java Week 6: Q2
- Java Week 6: Q3
- Java Week 6: Q4
- Java Week 6: Q5
- Feedback For Week 6

Week 7 :

Week 8 :

Week 9 :

Week 10 :

Week 11 :

Week 12 :

Solution

DOWNLOAD VIDEOS

Text Transcripts

Programming Test - (April 11 - 10AM - 12 PM)

Programming Test - (April 11 - 8PM - 10 PM)

Execution of two or more threads occurs in a random order. The keyword 'synchronized' in Java is used to control the execution of thread in a strict sequence. In the following, **the program is expected to print some numbers**. Use 'synchronized' keyword, so 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

Test Case 1

Input Expected Output Actual Output Status

```
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 2020-10-21, 15:55 IST

Your last recorded submission was :

```
1 class Execute{
2     synchronized void print(int n)
3     {
4         for(int i=1;i<=5;i++)
5         {
6             System.out.println(n*i);
7             try
8             {
9                 Thread.sleep(400);
10            }
11            catch(Exception e)
12            {
13                System.out.println(e);
14            }
15        }
16    }
17 } // Ending Execute class
18
19 class Thread1 extends Thread{
20     Execute t;
21     Thread1(Execute t){
22         this.t=t;
23     }
24     public void run(){
25         t.print(5);
26     }
27 }
28
29 class Thread2 extends Thread{
30     Execute t;
31     Thread2(Execute t){
32         this.t=t;
33     }
34     public void run(){
35         t.print(100);
36     }
37 }
38
39 public class Question64{
40     public static void main(String args[]){
41         Execute obj = new Execute();//only one object
42         Thread1 t1=new Thread1(obj);
43         Thread2 t2=new Thread2(obj);
44         t1.start();
45         t2.start();
46     }
47 }
48
```

Sample solutions (Provided by instructor)

```
1 class Execute{
2     // Just add 'synchronized' in the method
3     synchronized void print(int n){
4         for(int i=1;i<=5;i++){
5             System.out.println(n*i);
6             try{
7                 Thread.sleep(400);
8             }catch(Exception e){
9                 System.out.println(e);
10            }
11        }
12    }
13 }
```

```

10     }
11 }
12 }
13 }
14 } // Ending Execute class
15
16 class Thread1 extends Thread{
17     Execute t;
18     Thread1(Execute t){
19         this.t=t;
20     }
21     public void run(){
22         t.print(5);
23     }
24 }
25
26 class Thread2 extends Thread{
27     Execute t;
28     Thread2(Execute t){
29         this.t=t;
30     }
31     public void run(){
32         t.print(100);
33     }
34 }
35
36 public class Question64{
37     public static void main(String args[]){
38         Execute obj = new Execute(); //only one object
39         Thread1 t1=new Thread1(obj);
40         Thread2 t2=new Thread2(obj);
41         t1.start();
42         t2.start();
43     }
44 }
45

```