4 Nov 2020 Rinoy Kuriyakose R3 56

**Experiment: 14**

**Aim:**

|  |  |
| --- | --- |
|  | Write a Java program that implements a multi-threaded program which has three threads. First thread generates a random integer every 1 second. If the value is even, the second thread computes the square of the number and prints. If the value is odd the third thread will print the value of the cube of the number |

**Concept Used:**

Multithreading, Random generator.

**Algorithm:**

Class Thread1 extends Thread

1. import java.util.Random

2. Initialise Random object rand and static field random

3. Override public void run() method

4. num = rand.nextInt(100)

5. Print num

6. Thread2 obj2 = new Thread2(num)

7. Thread3 obj3 = new Thread3(num)

8. if(random%2==0)

9. obj2.start()

10. else

11. obj3.start()

12.endif

Class Thread2 extends Thread

1. Override public void run() method

2. Print Math.pow(num, 2)

3. End run()

Class Thread3 extends Thread

1. Override public void run() method

2. Print Math.pow(num, 3)

3. End run()

Class ThreadMain

1. for i=0 till 5

2. Thread1 obj1 = new Thread1()

3. obj1.start()

4. obj1.sleep(1000)

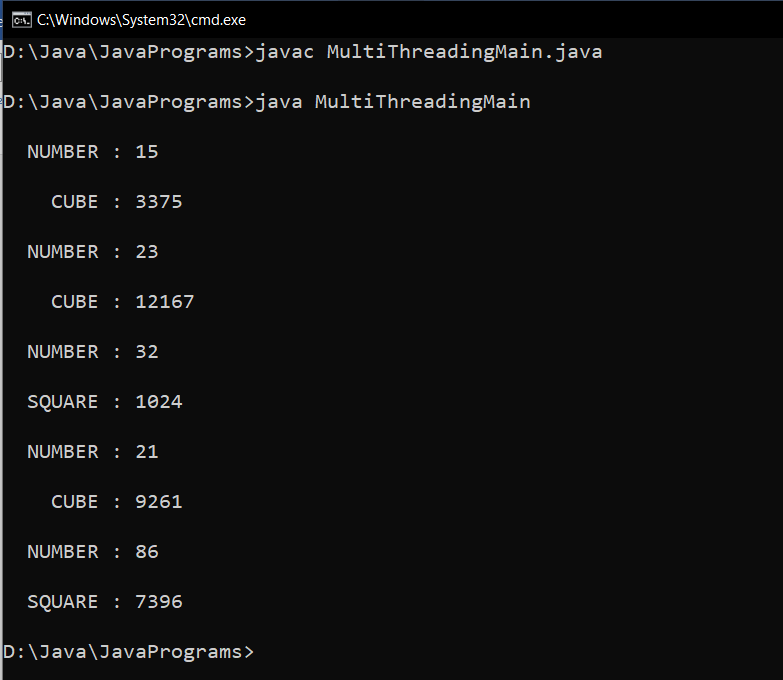
5. endfor

**Program:**

import java.util.\*;  
class Thread1 extends Thread

{ int num;  
 public void run(){  
 Random obj = new Random();  
 num=obj.nextInt(100);  
 System.out.println("\n NUMBER : "+num);  
 Thread2 obj2 = new Thread2(num);  
 Thread3 obj3 = new Thread3(num);  
 if(num%2==0){  
 obj2.start();  
 }else {  
 obj3.start();  
 }  
 }  
}  
class Thread2 extends Thread{  
 int num;  
 public void run(){  
 System.out.println("\n SQUARE : "+num\*num);  
 }  
 Thread2(int num) {  
 this.num=num;  
 }  
}  
class Thread3 extends Thread  
{  
 int num;  
 public void run(){  
 System.out.println("\n CUBE : "+num\*num\*num);  
 }  
 Thread3(int num){  
 this.num=num;  
 }  
}  
public class MultiThreadingMain {  
 public static void main(String[] args) throws InterruptedException{  
 for(int i=0;i<5;i++){  
 Thread1 obj1 = new Thread1();  
 obj1.start();  
 obj1.sleep(1000);  
 }  
 }  
}

**Output:**



**Result:**

Multi-threading is implemented in a Java program.

4 Nov 2020 Rinoy Kuriyakose R3 56

**Experiment: 15**

**Aim:**

Write a Java program that shows thread synchronization

**Concept Used:**

Multithreading, Thread synchronization.

**Algorithm:**

Class Number

1. synchronized print(n, )

2. for i = 1 to 5

3. Print n

4. endfor

5. End print()

class Thread1 extends Thread

1. Declare field num (object of class Number) and define a constructor to initialise num

2. public void run()

3. obj.print(1)

4. End run()

class Thread1 extends Thread

1. Declare field num (object of class Number) and define a constructor to initialise num

2. public void run()

3. obj.print(2)

4. End run()

class ThreadSynchronization

1. Initialise Number obj

2. Thread1 obj1 = new Thread1(obj)

3.Thread2 obj2 = new Thread2(obj);

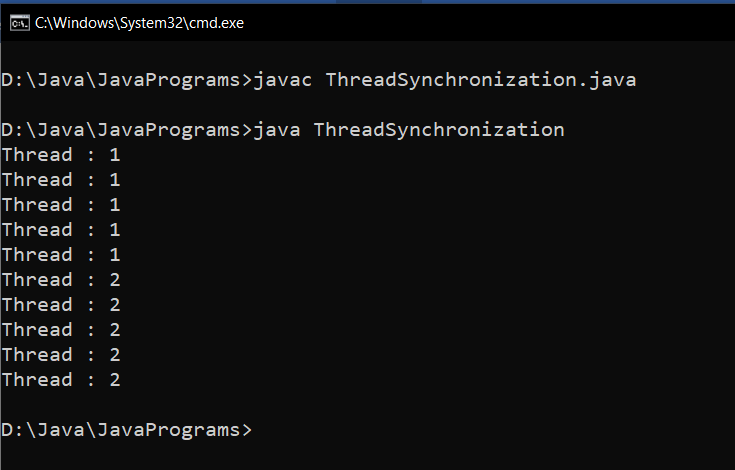
4. obj1.start();

5. obj2.start();

**Program:**

class Number{  
 synchronized void print(int n){  
 for(int i=1;i<=5;i++){  
 System.out.println("Thread : "+n);  
 }  
 }  
}  
class Thread1 extends Thread{  
 Number num;  
 Thread1(Number num){  
 this.num=num;  
 }  
 public void run(){  
 num.print(1);  
 }  
}  
class Thread2 extends Thread{  
 Number num;  
 Thread2(Number num){  
 this.num=num;  
 }  
 public void run(){  
 num.print(2);  
 }  
}  
  
public class ThreadSynchronization{  
 public static void main(String args[]){  
 Number obj = new Number();  
 Thread1 obj1 = new Thread1(obj);  
 Thread2 obj2 = new Thread2(obj);  
 obj1.start();  
 obj2.start();  
 }  
}

**Output:**



**Result:**

Thread synchronization is carried out in a Java program.