UNIT 3

ASSIGMENT 5

Name:- Sciddhanto Sinha
Roll No:-2213111
Division:- SY2

Assignment Title:

Multithreading in java

Aim:

Write a java program that implements a multi-thread application that has three threads. First thread generates a random integer every 1 second and 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.

Pre-Requisites: C/C++

Objective:

To achieve multithreading in java by implementing Runnable Interface or extending the Thread class.

Outcomes:

Students are able to Understand and Apply Multithreading concepts with java programming

Theory:

Multithreading is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of CPU. Each part of such program is called a thread. So, threads are light-weight processes within a process.

Threads can be created by using two mechanisms :

1. Extending the Thread class
2. Implementing the Runnable Interface

Thread creation by extending the Thread class

We create a class that extends the java.lang.Thread class. This class overrides the run() method available in the Thread class. A thread begins its life inside run() method. We create an object of our new class and call start() method to start the execution of a thread. Start() invokes the run() method on the Thread object.

```
// Java code for thread creation by extending

// the Thread class

class MultithreadingDemo extends Thread {

    public void run()

    {

        try {

            // Displaying the thread that is running

          System.out.println("Thread " + Thread.currentThread().getId() + " is
                                                  running");

        }

        catch (Exception e) {

            // Throwing an exception

            System.out.println("Exception is caught");

        }

    }

}
```

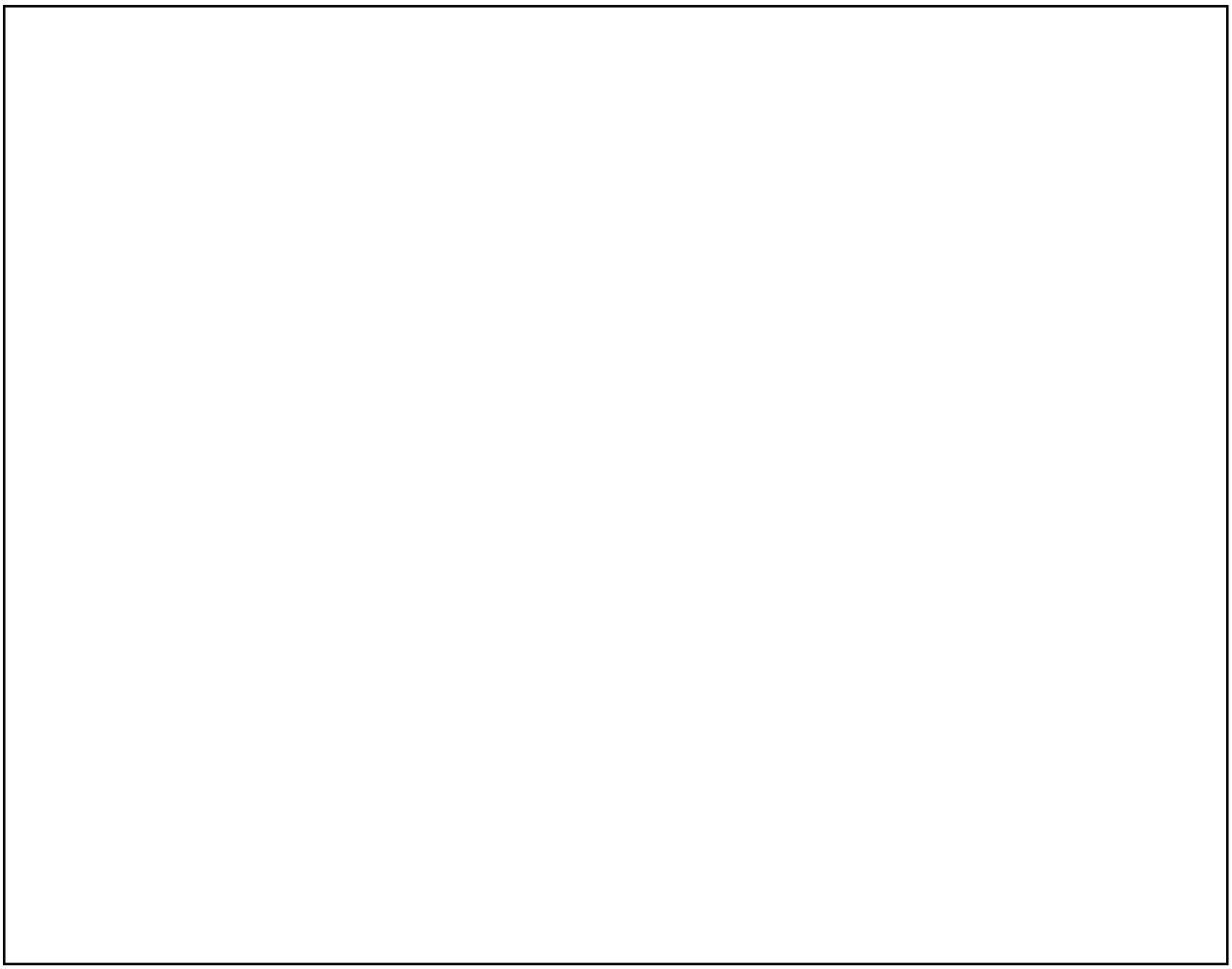
```
 // Main Class
public class Multithread {
    public static void main(String[] args)
    {
        int n = 8; // Number of threads
        for (int i = 0; i < n; i++) {
            MultithreadingDemo object = new MultithreadingDemo();
            object.start();
        }
    }
}
```

Output:

```
Thread 15 is running
Thread 14 is running
Thread 16 is running
Thread 12 is running
Thread 11 is running
Thread 13 is running
Thread 18 is running
Thread 17 is running
```

Thread creation by implementing the Runnable Interface

We create a new class which implements java.lang.Runnable interface and overrides run() method. Then we instantiate a Thread object and call start() method on this object.



```
// Java code for thread creation by implementing
// the Runnable Interface
class MultithreadingDemo implements Runnable {
    public void run()
    {
        try {
            // Displaying the thread that is running
   System.out.println( "Thread " + Thread.currentThread().getId()
                                    + " is running");
        }
        catch (Exception e) {
            // Throwing an exception
            System.out.println("Exception is caught");
        }
    }
}
 // Main Class
class Multithread {
    public static void main(String[] args)
    {
```

```
        int n = 8; // Number of threads
        for (int i = 0; i < n; i++) {
            Thread object = new Thread(new MultithreadingDemo());
            object.start();
        }
    }
}
```

Output

```
Thread 13 is running
Thread 11 is running
Thread 12 is running
Thread 15 is running
Thread 14 is running
Thread 18 is running
Thread 17 is running
Thread 16 is running
```

Algorithm/Steps:

1. Start the program

2. Create a class with the name “even implements Runnable” and “odd implementsRunnable”.

3. Create thread objects and Random class objects.

4. Pass the objects of our class to the Thread class.

5. Call the start method.

Conclusion:

Hence, we studied multithreading in java.

Frequently Asked Questions:

1. Define Thread class and Runnable interface?
2. What are thread methods?
3. What is alive method?
4. What is thread interrupt?
5. What are the states associated in the thread?
6. What is the difference between process and thread?

CODE: -

```
package unit3;
import java.util.Random;

class RandomNumberThread1 extends Thread{
    public void run()
    {
        Random random=new Random();
        System.out.println();
        System.out.println("_____Random
Number_____");
        for(int i=0;i<10;i++)
        {
            int randomInt=random.nextInt(100);
            System.out.println("Random numbers :: "+randomInt);
            if (randomInt%2==0)
            {


                SquareThread1 st=new SquareThread1(randomInt);
                st.start();
            }
            else {
```

```
                CubeThread1 ct=new CubeThread1(randomInt);
                ct.start();
            }
            try{
                Thread.sleep(1000);
            }
            catch (InterruptedException ex){
                System.out.println("Error"+ex);
            }
        }
    }
}
class SquareThread1 extends Thread{
    int num;
    SquareThread1(int rn)
    {
        num=rn;
    }
    public void run()
    {

        System.out.println("Square of "+num+" is "+num*num);
    }

}
class CubeThread1 extends Thread{
    int num;
    CubeThread1(int rn)
    {
        num=rn;
    }
    public void run()
    {

        System.out.println("Cube of "+num+" is "+num*num*num);
    }
}
public class pc1 {
    public static void main(String[] args) {

        RandomNumberThread1 rnt=new RandomNumberThread1();
        rnt.start();

    }
}
```

OUTPUT: -

```
_____________________________________Random Number_____________________________________
Random numbers :: 38
Square of 38 is 1444
Random numbers :: 3
Cube of 3 is 27
Random numbers :: 48
Square of 48 is 2304
Random numbers :: 9
Cube of 9 is 729
Random numbers :: 52
Square of 52 is 2704
Random numbers :: 24
Square of 24 is 576
Random numbers :: 75
Cube of 75 is 421875
Random numbers :: 64
Square of 64 is 4096
Random numbers :: 3
Cube of 3 is 27
Random numbers :: 35
Cube of 35 is 42875

Process finished with exit code 0
```