Assignment – 7

1. Write a Java program in which a total of 4 threads should run. Set different priorities for the thread.

```java
public class demo {
  public static void main(String[] args) {
    System.out.println("Name : Gokul Sarkar \nRoll No : 46");
    Thread thread1 = new Thread(new MyRunnable(), "Thread 1");
    Thread thread2 = new Thread(new MyRunnable(), "Thread 2");
    Thread thread3 = new Thread(new MyRunnable(), "Thread 3");
    Thread thread4 = new Thread(new MyRunnable(), "Thread 4");

    thread1.setPriority(Thread.MIN_PRIORITY);
    thread2.setPriority(Thread.NORM_PRIORITY);
    thread3.setPriority(Thread.NORM_PRIORITY);
    thread4.setPriority(Thread.MAX_PRIORITY);

    thread1.start();
    thread2.start();
    thread3.start();
    thread4.start();
 }
 private static class MyRunnable implements Runnable {
   public void run() {
     System.out.println(" " + Thread.currentThread().getName() + " is
running with priority " + Thread.currentThread().getPriority());
    }
  }
}
```
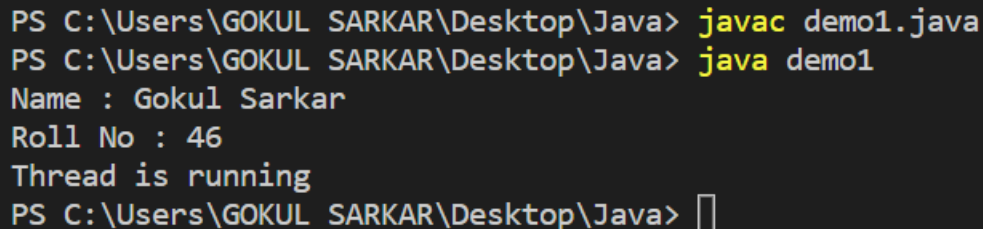
Output:

```
PS C:\Users\GOKUL SARKAR\Desktop\Java> javac demo.java
PS C:\Users\GOKUL SARKAR\Desktop\Java> java demo
Name : Gokul Sarkar
Roll No : 46
 Thread 3 is running with priority 5
 Thread 4 is running with priority 10
 Thread 2 is running with priority 5
 Thread 1 is running with priority 1
PS C:\Users\GOKUL SARKAR\Desktop\Java> []
```

2. Write a Java Program to Create a Thread that Implements the Runnable Interface.

```java
class MyRunnable implements Runnable {
  public void run() {
    System.out.println("Thread is running");
  }
}
public class demo1 {
  public static void main(String[] args) {
    System.out.println("Name : Gokul Sarkar \nRoll No : 46");
    MyRunnable myRunnable = new MyRunnable();

    Thread myThread = new Thread(myRunnable);
    myThread.start();
  }
}
```
Output:

```
PS C:\Users\GOKUL SARKAR\Desktop\Java> javac demo1.java
PS C:\Users\GOKUL SARKAR\Desktop\Java> java demo1
Name : Gokul Sarkar
Roll No : 46
Thread is running
PS C:\Users\GOKUL SARKAR\Desktop\Java> []
```

3. Write a Java Program to Check the Priority Level of a Thread.

```java
public class demo2 {
  public static void main(String[] args) {
    System.out.println("Name : Gokul Sarkar \nRoll No : 46");
    Thread myThread = new Thread();
    int priority = myThread.getPriority();
    System.out.println("Thread priority: " + priority);
  }
}
```

Output:

```
PS C:\Users\GOKUL SARKAR\Desktop\Java> javac demo2.java
PS C:\Users\GOKUL SARKAR\Desktop\Java> java demo2
Name : Gokul Sarkar
Roll No : 46
Thread priority: 5
PS C:\Users\GOKUL SARKAR\Desktop\Java> []
```

4. Write a Java Program Defining Thread By Extending the Thread class.

```java
class MyThread extends Thread {
   public void run() {
      System.out.println("Thread is running");
   }
}
public class demo3 {
   public static void main(String[] args) {
      System.out.println("Name : Gokul Sarkar \nRoll No : 46");
      MyThread myThread = new MyThread();
      myThread.start();
   }
}
```

Output:

```
PS C:\Users\GOKUL SARKAR\Desktop\Java> javac demo3.java
PS C:\Users\GOKUL SARKAR\Desktop\Java> java demo3
Name : Gokul Sarkar
Roll No : 46
Thread is running
PS C:\Users\GOKUL SARKAR\Desktop\Java> []
```

5. Write a Java Program to Get the Name of a Running Thread.

```java
public class demo4 {
   public static void main(String[] args) {
      Thread thread = new Thread() {
          public void run() {
         String threadName = Thread.currentThread().getName();
```

```
        System.out.println("Thread name: " + threadName);
        }
    };
    thread.start();
    String mainThreadName = Thread.currentThread().getName();
    System.out.println("Main thread name: " + mainThreadName);
    System.out.println("Name : Gokul Sarkar \nRoll No : 46");
  }
}
```

Output:

```
PS C:\Users\GOKUL SARKAR\Desktop\Java> javac demo4.java
PS C:\Users\GOKUL SARKAR\Desktop\Java> java demo4
Thread name: Thread-0
Main thread name: main
Name : Gokul Sarkar
Roll No : 46
PS C:\Users\GOKUL SARKAR\Desktop\Java>
```

6. Write a Java Program to Stop a Thread.

```
public class demo5 {
   public static void main(String[] args) throws InterruptedException {
      MyThread myThread = new MyThread();
      myThread.start();

      Thread.sleep(5000);
      myThread.interrupt();
   }
}
class MyThread extends Thread {
   public void run() {
      while (!isInterrupted()) {
         System.out.println("Thread is running...");
         System.out.println("Name : Gokul Sarkar  Roll No : 46");
      try {
         Thread.sleep(1000); // Sleep for 1 second
      }
      catch (InterruptedException e) {
         return;
```

Name : Gokul Sarkar
Roll No : 46

```
      }
    }
  }
}
```
Output:

```
PS C:\Users\GOKUL SARKAR\Desktop\Java> javac demo5.java
PS C:\Users\GOKUL SARKAR\Desktop\Java> java demo5
Thread is running...
Name : Gokul Sarkar   Roll No : 46
Thread is running...
Name : Gokul Sarkar   Roll No : 46
Thread is running...
Name : Gokul Sarkar   Roll No : 46
Thread is running...
Name : Gokul Sarkar   Roll No : 46
Thread is running...
Name : Gokul Sarkar   Roll No : 46
PS C:\Users\GOKUL SARKAR\Desktop\Java> []
```

7. Write a Java Program to Check Whether Define a Thread Class Without Defining run( ) Method in the Class.

```
class MyThread extends Thread {
}
public class demo6 {
   public static void main(String[] args) {
      System.out.println("Name : Gokul Sarkar \nRoll No : 46");

      MyThread myThread = new MyThread();

      myThread.start();
   }
}
```
Output:

```
PS C:\Users\GOKUL SARKAR\Desktop\Java> javac demo6.java
PS C:\Users\GOKUL SARKAR\Desktop\Java> java demo6
Name : Gokul Sarkar
Roll No : 46
PS C:\Users\GOKUL SARKAR\Desktop\Java> []
```

8. Write a Java Program to Show that Method Will be Verified Whether it is Synchronized or Not.Define a Thread Class Without Defining run() Method in the Class.

```java
public class Main {
  public synchronized void synchronizedMethod() {
    System.out.println("This method is synchronized");
  }
  public void nonSynchronizedMethod() {
    System.out.println("This method is not synchronized");
  }
  public static void main(String[] args) {
    System.out.println("Name : Gokul Sarkar \nRoll No : 46");
    Main main = new Main();
    main.synchronizedMethod();
    main.nonSynchronizedMethod();
    Thread t1 = new Thread(() -> {
      main.synchronizedMethod();
    });
    Thread t2 = new Thread(() -> {
      main.synchronizedMethod();
    });
    t1.start();
    t2.start();
    Thread t3 = new Thread(() -> {
      main.nonSynchronizedMethod();
    });
    Thread t4 = new Thread(() -> {
      main.nonSynchronizedMethod();
    });
    t3.start();
    t4.start();
  }
}
```
Output:

```
Name : Gokul Sarkar
Roll No : 46
This method is synchronized
This method is not synchronized
This method is synchronized
This method is synchronized
This method is not synchronized
This method is not synchronized


...Program finished with exit code 0
Press ENTER to exit console.
```

9. Create 4 threads with priority 1,3,5,7 respectively. Update a counter in each of the threads for 10 ms. Print the final value of the count for each thread.
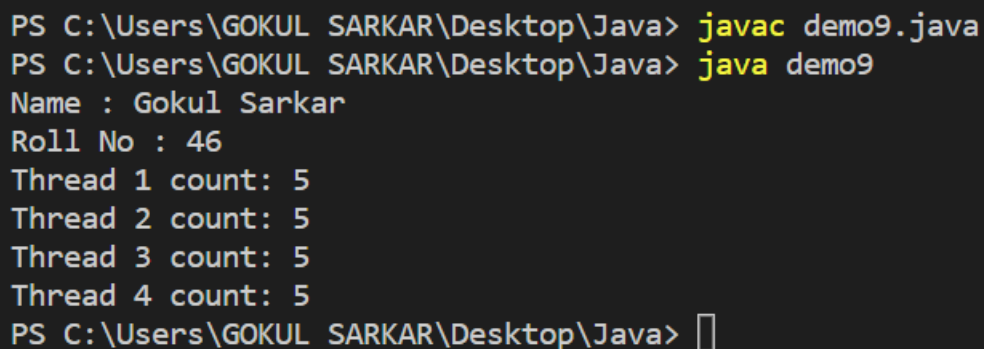
```java
public class demo9 {
  public static void main(String[] args) {
    CounterThread t1 = new CounterThread("Thread 1", 1);
    CounterThread t2 = new CounterThread("Thread 2", 3);
    CounterThread t3 = new CounterThread("Thread 3", 5);
    CounterThread t4 = new CounterThread("Thread 4", 7);
    t1.start();
    t2.start();
    t3.start();
    t4.start();
    try {
      t1.join();
      t2.join();
      t3.join();
      t4.join();
    } catch (InterruptedException e) {
      System.out.println("Main thread interrupted.");
    }
    System.out.println("Name : Gokul Sarkar \nRoll No : 46");
    System.out.println(t1.getName() + " count: " + t1.getCount());
    System.out.println(t2.getName() + " count: " + t2.getCount());
    System.out.println(t3.getName() + " count: " + t3.getCount());
    System.out.println(t4.getName() + " count: " + t4.getCount());
  }
```

```java
}
class CounterThread extends Thread {
   private int count = 0;
   public CounterThread(String name, int priority) {
      super(name);
      setPriority(priority);
   }
   public int getCount() {
      return count;
   }
   public void run() {
      for (int i = 0; i < 5; i++) {
         count++;
         try {
            Thread.sleep(10);
         } catch (InterruptedException e) {
            System.out.println(getName() + " interrupted.");
         }
      }
   }
}
```

Output:

```
PS C:\Users\GOKUL SARKAR\Desktop\Java> javac demo9.java
PS C:\Users\GOKUL SARKAR\Desktop\Java> java demo9
Name : Gokul Sarkar
Roll No : 46
Thread 1 count: 5
Thread 2 count: 5
Thread 3 count: 5
Thread 4 count: 5
PS C:\Users\GOKUL SARKAR\Desktop\Java> []
```

10.Write a Java Program to Use Method Level Synchronization.

```java
class Counter {
   private int count;
   public synchronized void increment() {
      count++;
   }
   public int getCount() {
      return count;
```

```java
    }
}
class CountThread extends Thread {
    private Counter counter;
    private int times;
    public CountThread(Counter counter, int times) {
        this.counter = counter;
        this.times = times;
    }
    public void run() {
        for (int i = 0; i < times; i++) {
            counter.increment();
        }
    }
}
public class demo8 {
    public static void main(String[] args) {
        Counter counter = new Counter();
        CountThread thread1 = new CountThread(counter, 10000);
        CountThread thread2 = new CountThread(counter, 20000);
        thread1.start();
        thread2.start();
        try {
            thread1.join();
            thread2.join();
        } catch (InterruptedException e) {
            System.out.println("Thread interrupted");
        }
        System.out.println("Name : Gokul Sarkar \nRoll No : 46");
        System.out.println("Final count: " + counter.getCount());
    }
}
```

Output:

```
PS C:\Users\GOKUL SARKAR\Desktop\Java> javac demo8.java
PS C:\Users\GOKUL SARKAR\Desktop\Java> java demo8
Name : Gokul Sarkar
Roll No : 46
Final count: 30000
PS C:\Users\GOKUL SARKAR\Desktop\Java> []
```