Task 7: Writing Thread-Safe Code, Immutable Objects

Design a thread-safe Counter class with increment and decrement methods. Then demonstrate its usage from multiple threads. Also, implement and use an immutable class to share data between threads.

```
package com.wipro.model;

import java.util.concurrent.*;

class Counter {

private int count;

public Counter() {

this.count = 0;

}

public synchronized void increment() {

count++;

}

public synchronized void decrement() {

count--;

}

public synchronized int getCount() {

return count;

}

}

final class Data {

private final int value;

public Data(int value) {

this.value=value;

}

public int getValue() {

return value;

}

}

public class ThreadSafeDemo {

public static void main(String[] args)throws InterruptedException {
```

```
Counter counter=new Counter();
Thread thread1=new Thread(() -> {
for (int i=0;i<1000;i++) {
counter.increment();
}
});
Thread thread2=new Thread(()->{
for (int i=0;i<1000;i++) {
counter.decrement();
}
});
thread1.start();
thread2.start();
thread1.join();
thread2.join();
System.out.println("Final Count: "+counter.getCount());
Data data=new Data(42);
Thread thread3=new Thread(() -> {
System.out.println("Thread-3: "+data.getValue());
});
Thread thread4 = new Thread(()->{
System.out.println("Thread-4: "+data.getValue());
});
thread3.start();
thread4.start();
thread3.join();
thread4.join();
}
}
```