import java.util.concurrent.locks.ReentrantReadWriteLock;

class SharedResource {

    private String data = "Initial Data";

    private final ReentrantReadWriteLock lock = new ReentrantReadWriteLock();

    // Method for readers to read data

    public void readData(String readerName) {

        lock.readLock().lock();  // Acquire read lock

        try {

            System.out.println(readerName + " is reading the data: " + data);

            Thread.sleep(1000);  // Simulate reading

        } catch (InterruptedException e) {

            e.printStackTrace();

        } finally {

            lock.readLock().unlock();  // Release read lock

        }

    }

    // Method for writers to write data

    public void writeData(String writerName, String newData) {

        lock.writeLock().lock();  // Acquire write lock

        try {

            System.out.println(writerName + " is writing data...");

            Thread.sleep(1000);  // Simulate writing

            data = newData;

            System.out.println(writerName + " has written the data: " + data);

        } catch (InterruptedException e) {

            e.printStackTrace();

        } finally {

            lock.writeLock().unlock();  // Release write lock

        }

    }

}

class Reader extends Thread {

    private final SharedResource sharedResource;

    private final String readerName;

    public Reader(SharedResource sharedResource, String readerName) {

        this.sharedResource = sharedResource;

        this.readerName = readerName;

    }

    public void run() {

        sharedResource.readData(readerName);

    }

}

class Writer extends Thread {

    private final SharedResource sharedResource;

    private final String writerName;

    private final String newData;

    public Writer(SharedResource sharedResource, String writerName, String newData) {

        this.sharedResource = sharedResource;

        this.writerName = writerName;

        this.newData = newData;

    }

    public void run() {

        sharedResource.writeData(writerName, newData);

    }

}

public class ReaderWriterProblem {

    public static void main(String[] args) {

        SharedResource sharedResource = new SharedResource();

        // Creating reader and writer threads

        Reader reader1 = new Reader(sharedResource, "Reader 1");

        Reader reader2 = new Reader(sharedResource, "Reader 2");

        Writer writer1 = new Writer(sharedResource, "Writer 1", "New Data 1");

        Writer writer2 = new Writer(sharedResource, "Writer 2", "New Data 2");

        // Starting threads

        reader1.start();

        writer1.start();

        reader2.start();

        writer2.start();

    }

}