

廈門大學



信息学院软件工程系

《JAVA 程序设计》实验报告

实验十四

姓名：庾晓萍

学号：20420192201952

学院：信息学院

专业：软件工程

完成时间：2022/5/29

一、实验目的及要求

（一）实验目的

- 1、熟悉多线程编程
- 2、熟悉网络编程

（二）实验要求

- 1、按照题目要求写代码和实验报告，并上传到 FTP

二、实验题目及实现过程

一、基本题目：

请完成一个多线程的程序。

（一）实验环境

操作系统：Windows 10；

IDE：Eclipse Java 2018-12

编程语言：Java；

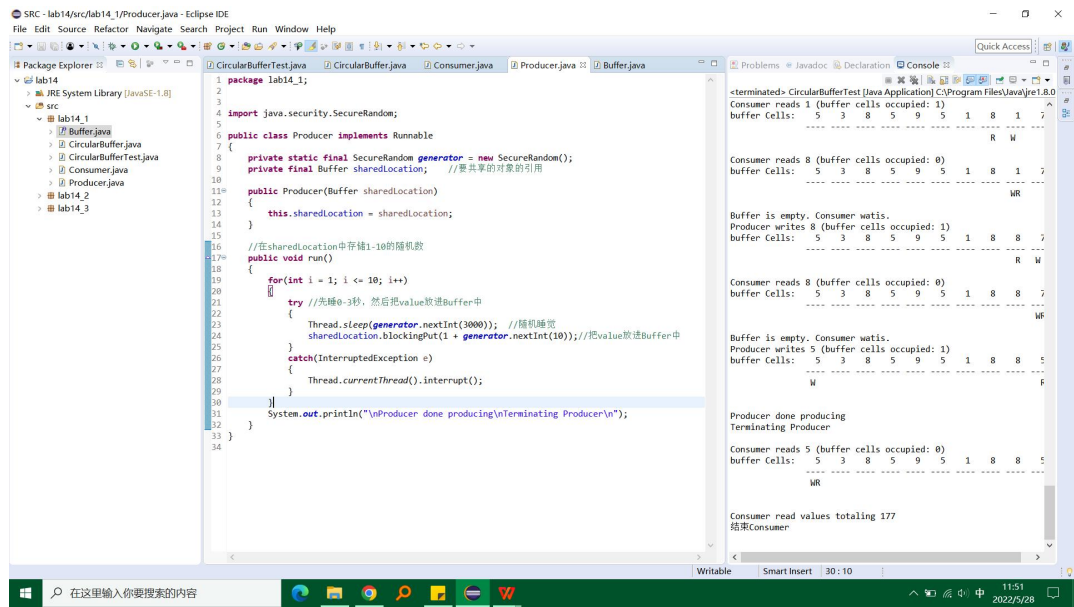
（二）实现过程

（1）设计思路

- 三个线程随机产生 1-10 的随机数放入一个包含 10 个元素的循环缓冲区，一个线程从里面取出元素并输出。其中循环缓冲区用自己定义的类。
- 在 **Buffer** 接口中定义了要被 **Producer** 和 **Consumer** 调用的方法，其中 **blockingPut** 将 **value** 放到 **Buffer** 中，而 **blockingGet** 从 **Buffer** 中返回值。
- **Consumer** 继承了 **Runnable** 类，实现 **Runnable** 类中的抽象方法 **run**。从 **sharedLocation** 中读 30 次，并把读到的 **value** 累加起来，每次读时，先用 **Thread.sleep** 让线程休息，再用 **Buffer** 接口中 **blockingGet** 方法获取 **value** 加入 **sum** 中。读取 30 后输出 **sum**。
- **Producer** 类同样继承了 **Runnable** 类。其中 **sharedLocation** 是 **Buffer** 类的实例，也是要共享的对象的引用。实现 **Runnable** 中的抽象方法 **run**，在 **sharedLocation** 中存储 1-10 的随机数。先随机睡觉，然后把 **value** 放入 **Buffer** 中。
- 自己定义一个类实现循环缓冲区，用一个有三个元素的 **buffer** 数组来实现。**occupiedCells** 用来数有几个 **buffers** 被暂用，**writeIndex** 代表下一个要写的元素的下标。**readIndex** 代表下一个要读的元素的下标。**displayState** 方法用于展示 **buffer** 的相关状态。同步方法 **blockingPut** 和 **blockingGet** 用与读写 **buffer**。对于方法 **blockingPut**，一直等到 **buffer** 有空位时再写入数据，当没有空位的时候，就把线程放入锁定状态。而对于方法 **blockingGet**，等到有数据的时候再读，没数据就等。
- 在 **CircularBufferTest** 方法中，建立一个 **ExecutorService** 对象，作为新的线程池。执行三个 **Producer** 线程，和一个 **Consumer** 线程，最后用 **shutdown** 方法关闭线程池。

（三） 过程截图

（1）全屏截图



```
package lab14_1;

import java.security.SecureRandom;

public class Producer implements Runnable {
    private static final SecureRandom generator = new SecureRandom();
    private final Buffer sharedLocation; // 要共享的对象的引用

    public Producer(Buffer sharedLocation) {
        this.sharedLocation = sharedLocation;
    }

    // 在sharedLocation中存储1-10的随机数
    public void run() {
        for(int i = 1; i <= 10; i++) {
            try { // 先睡0-3秒, 然后把value放进Buffer中
                Thread.sleep(generator.nextInt(3000)); // 随机睡觉
                sharedLocation.blockingPut(1 + generator.nextInt(10)); // 把value放进Buffer中
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();
            }
        }
        System.out.println("Producer done producing\nTerminating Producer\n");
    }
}
```

```
<terminated> CircularBufferTest [Java Application] C:\Program Files\Java\jre1.8.0
Consumer reads 1 (buffer cells occupied: 1)
buffer Cells: 5 3 8 5 9 5 1 8 1 7
.....
R W

Consumer reads 8 (buffer cells occupied: 0)
buffer Cells: 5 3 8 5 9 5 1 8 1 7
.....
WR

Buffer is empty, Consumer waits.
Producer writes 8 (buffer cells occupied: 1)
buffer Cells: 5 3 8 5 9 5 1 8 8 7
.....
R W

Consumer reads 8 (buffer cells occupied: 0)
buffer Cells: 5 3 8 5 9 5 1 8 8 7
.....
WR

Buffer is empty, Consumer waits.
Producer writes 5 (buffer cells occupied: 1)
buffer Cells: 5 3 8 5 9 5 1 8 8 5
.....
W F

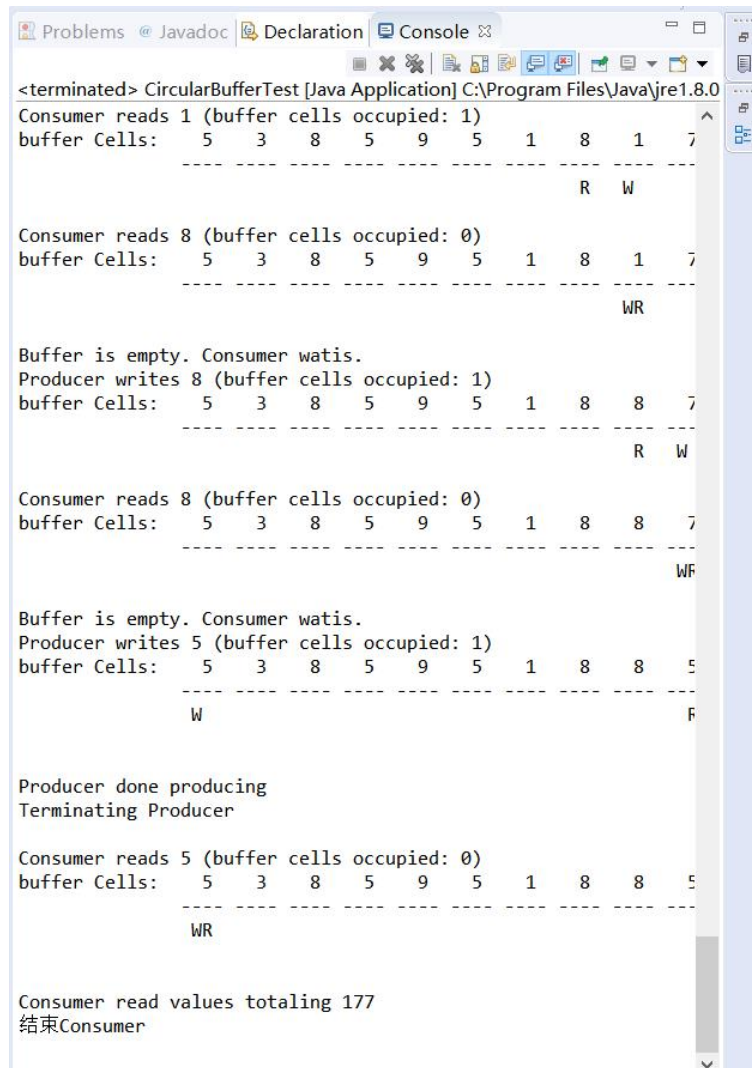
Producer done producing
Terminating Producer

Consumer reads 5 (buffer cells occupied: 0)
buffer Cells: 5 3 8 5 9 5 1 8 8 5
.....
WR

Consumer read values totaling 177
结束Consumer
```

(2) 运行结果

三个线程随机产生 1-10 的随机数放入一个包含 10 个元素的循环缓冲区,一个线程从里面取出元素并输出。其中循环缓冲区用自己定义的类。



```
<terminated> CircularBufferTest [Java Application] C:\Program Files\Java\jre1.8.0
Consumer reads 1 (buffer cells occupied: 1)
buffer Cells:  5  3  8  5  9  5  1  8  1  7
               -----
                           R  W

Consumer reads 8 (buffer cells occupied: 0)
buffer Cells:  5  3  8  5  9  5  1  8  1  7
               -----
                           WR

Buffer is empty. Consumer watis.
Producer writes 8 (buffer cells occupied: 1)
buffer Cells:  5  3  8  5  9  5  1  8  8  7
               -----
                           R  W

Consumer reads 8 (buffer cells occupied: 0)
buffer Cells:  5  3  8  5  9  5  1  8  8  7
               -----
                           WR

Buffer is empty. Consumer watis.
Producer writes 5 (buffer cells occupied: 1)
buffer Cells:  5  3  8  5  9  5  1  8  8  5
               -----
               W                                     F

Producer done producing
Terminating Producer

Consumer reads 5 (buffer cells occupied: 0)
buffer Cells:  5  3  8  5  9  5  1  8  8  5
               -----
               WR

Consumer read values totaling 177
结束Consumer
```

题目 2：修改第 1 题

（一）实验环境

操作系统：Windows 10;

IDE：Eclipse Java 2018-12

编程语言：Java;

(二) 实现过程

(1) 设计思路

- 采用 API 中已有的类替换循环缓冲区类定义, 并对程序做相应修改。
- 在 CircularBuffer 类中, 定义一个 ReentrantLock 类的 accessLock 对象作为用来控制同步 Buffer 的锁。canWrite 和 canRead 作为用来控制写、读的 condition。Condition 依赖于 Lock 接口, 生成一个 Condition 的基本代码是 lock.newCondition() (accessLock.newCondition)。对于 blockingPut 方法, 调用时首先让 accessLock 上锁, 一直等到 buffer 有空位, 再写入数据, 当没有空位的时候, 就通过 canWrite.await 等待。把线程放入锁定状态。等写操作完成后再开启读操作。

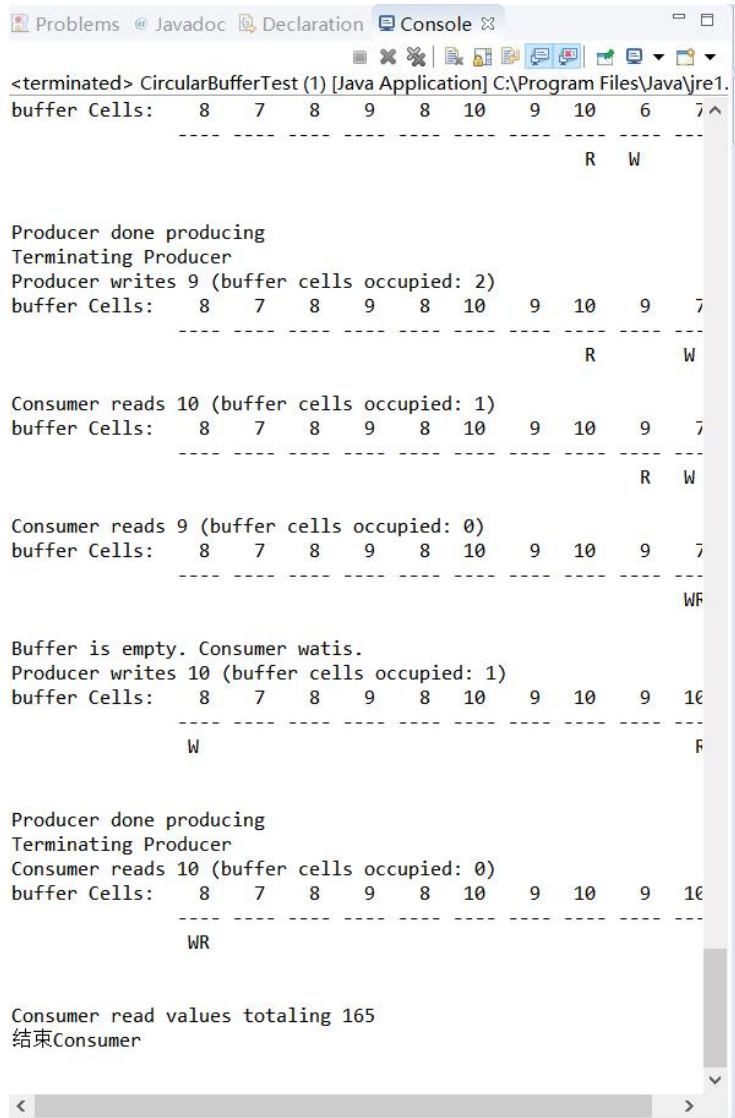
(三) 过程截图

(1) 全屏截图

The screenshot shows the Eclipse IDE with the following components:

- Package Explorer:** Shows the project structure with packages lab14, lab14.1, and lab14.2. The file CircularBufferTest.java is selected.
- Editor:** Displays the code for CircularBufferTest.java. The code implements a circular buffer using ReentrantLock and Condition. It includes methods for writing (blockingPut) and reading (blockingGet) data from the buffer.
- Console:** Shows the output of the program. It displays the state of the buffer (occupied cells, free cells) and the sequence of operations performed by the producer and consumer threads. The output shows the buffer being filled and then emptied, with the consumer waiting for data to be available.

(2) 运行结果



```
<terminated> CircularBufferTest (1) [Java Application] C:\Program Files\Java\jre1.
buffer Cells:  8  7  8  9  8  10  9  10  6  7 ^
-----
R  W

Producer done producing
Terminating Producer
Producer writes 9 (buffer cells occupied: 2)
buffer Cells:  8  7  8  9  8  10  9  10  9  7
-----
R  W

Consumer reads 10 (buffer cells occupied: 1)
buffer Cells:  8  7  8  9  8  10  9  10  9  7
-----
R  W

Consumer reads 9 (buffer cells occupied: 0)
buffer Cells:  8  7  8  9  8  10  9  10  9  7
-----
WF

Buffer is empty. Consumer watis.
Producer writes 10 (buffer cells occupied: 1)
buffer Cells:  8  7  8  9  8  10  9  10  9  10
-----
W  F

Producer done producing
Terminating Producer
Consumer reads 10 (buffer cells occupied: 0)
buffer Cells:  8  7  8  9  8  10  9  10  9  10
-----
WR

Consumer read values totaling 165
结束Consumer
```

题目 3：修改 TicTacToe 程序，补充“判断游戏结束”部分代码。

（一）实验环境

操作系统：Windows 10;

IDE：Eclipse Java 2018-12

编程语言：Java;

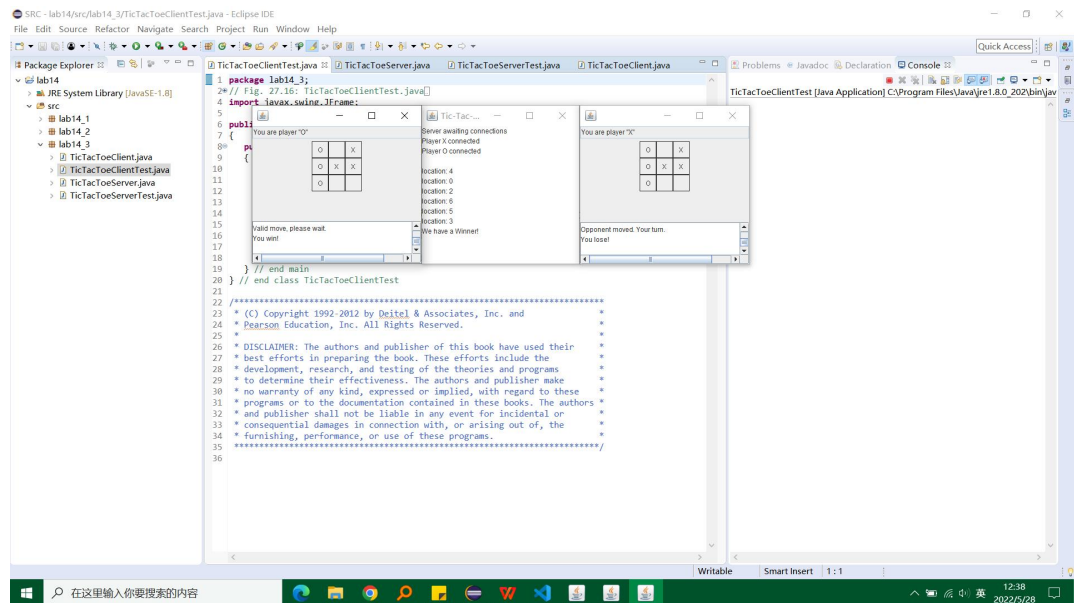
(二) 实现过程

(1) 实验思路

修改 TicTacToe 程序，补充“判断游戏结束”部分代码。

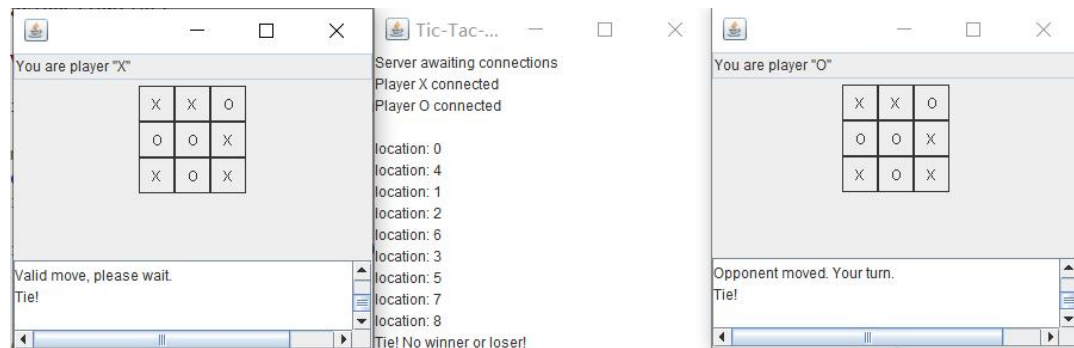
(三) 过程截图

(1) 全屏截图

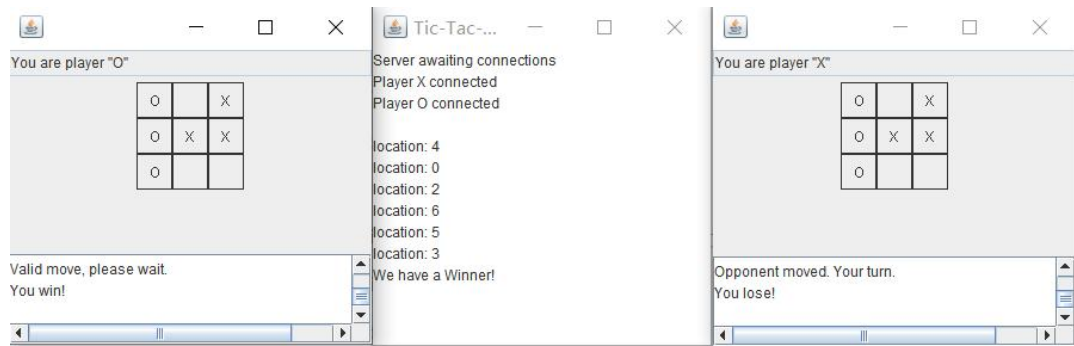


(2) 运行结果

① 平局



② 分出胜负



三、实验总结与心得记录

在本次实验过程中,我熟悉了 java 的语法,体会到了 JAVA 语言的优点。