**线程安全上机报告**

1. **代码（修改版）**

**在同一个文件夹中。**

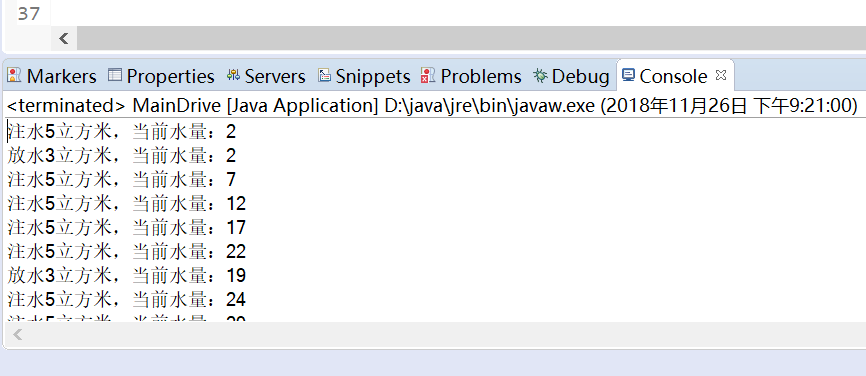
**二．测试过程以及结果和分析**

1. **只有俩个线程时候，改变放水和注水的休眠时间**

**注水：TimeUnit.*SECONDS*.sleep(1); 每次注入水量5**

**放水：TimeUnit.*SECONDS*.sleep(5); 每次放水量3**

**结果：**

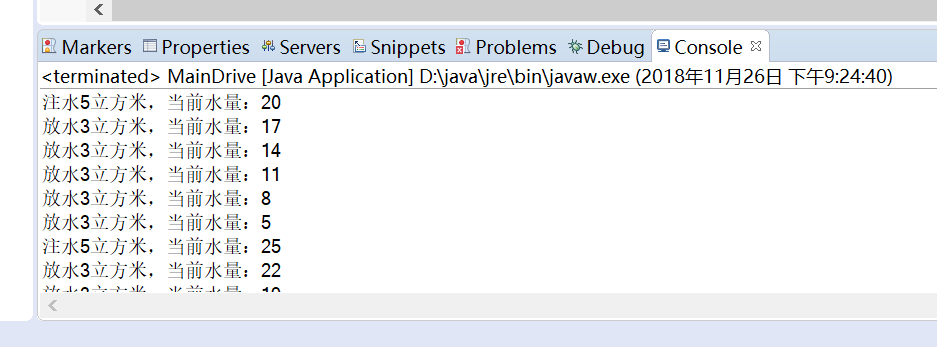


**分析：当只有俩个线程一开始时，是共同争夺资源的，谁先占用不一定。第一次放水放完了的时候将休眠5秒，是注水线程休眠时间的5倍，这个时候只有注水线程在注水，并且是连续注水5次。**

**注水： TimeUnit.*SECONDS*.sleep(5); 每次注水量为20**

**放水： TimeUnit.*SECONDS*.sleep(1); 每次放水量为3**

**结果：**



**分析：与第一个相反，注水休眠的时间是放水休眠的时间的5倍，因此可以从结果当中看到也是相反的，一开始，线程争夺到资源的概率是同等的。**

**2.改变线程的个数**

**改变1：注水线程3个，放水线程2个，如下：**

InputThread thread1 = **new** InputThread(queue); //注水

InputThread thread2 = **new** InputThread(queue);

InputThread thread3= **new** InputThread(queue);

OutputThread thread4 = **new** OutputThread(queue);//放水

OutputThread thread5 = **new** OutputThread(queue);

List<Callable<Object>>threadList=**new** ArrayList<>(5);

threadList.add(thread1);

threadList.add(thread2);

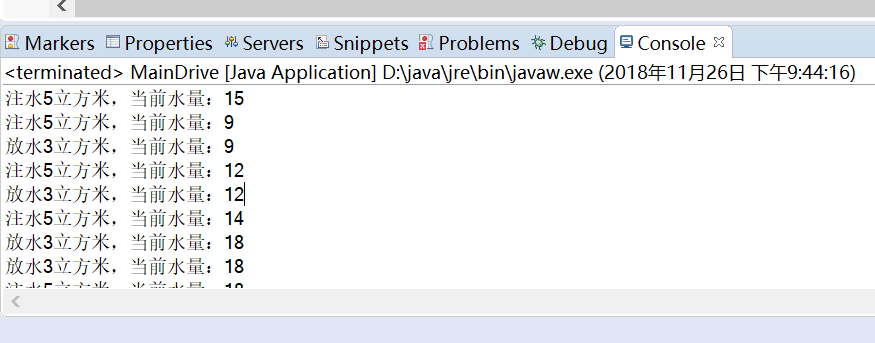
threadList.add(thread3);

threadList.add(thread4);

threadList.add(thread5);

**注水和放水的休眠时间均为1秒，每次注水量均为5，放水量均为1**

**结果：**

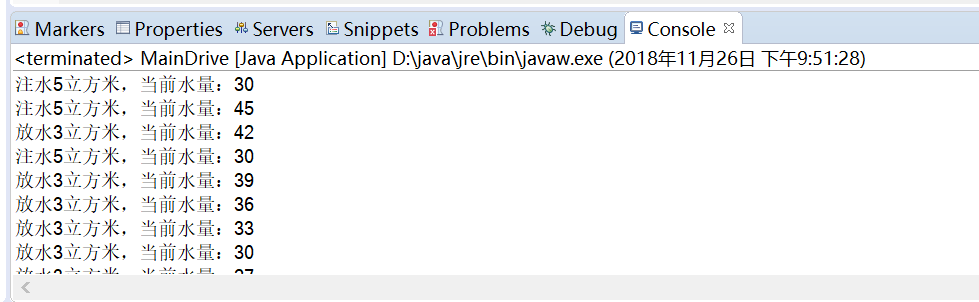


**分析：从结果当中可以看到，在睡眠时间相同的情况下，同类功能线程多的（注水）抢夺到资源的概率大一些。**

**改变2：线程个数、比例不变，调整注水和放水的休眠时间。**

**注水：TimeUnit.*SECONDS*.sleep(5); 每次注水量15**

**放水：TimeUnit.*SECONDS*.sleep(1); 每次放水量3**



**分析：在一开始时，同功能线程多（注水）的在抢夺资源的时候还是占优势，但是通过改变其休眠时间比例时，从结果当中可以看出，在某种程度上可以降低这种同种线程多的优势。**

1. **实验总结和感悟**
2. **进一步了解线程的机制，以及线程池的基本应用和方法，以及休眠时间、时间类、日期类、阻塞队列BlockingQueue、生产者和消费者模式。**
3. **在Java中带有返回值的接口线程，必须实现Callable接口，无返回值任务的线程必须实现Runnable接口，Callable并不可以直接运行的，需要Future Task的帮助，将Callable放入Future当中进行管理。**
4. **线程池(ExceutorService)，是将多个线程放在一个地方进行管理，批量添加线程，和future一起使用。ExceutorService定义了俩个批量执行任务的方法invokeAll()和ininvokeAny()。在批量执行或多选一的业务场景当中比较方便。Invokeall（）在所有任务都完成（包括成功、被中断、超时）后才返回，invokeany（）在任意一个任务成功（Exceutor被中断或者超时）后就会返回。**
5. **查阅的相关资料如下链接：**

[**https://www.cnblogs.com/tjudzj/p/4454490.html**](https://www.cnblogs.com/tjudzj/p/4454490.html)

[**https://www.cnblogs.com/mada0/p/4713181.html**](https://www.cnblogs.com/mada0/p/4713181.html)

[**https://www.cnblogs.com/LCX/p/4509610.html**](https://www.cnblogs.com/LCX/p/4509610.html)

**http://www.importnew.com/27482.html**