## 第二次实验完成项目

本次实验一共5道题目，其实本质上的关系还是很大的，有很多共性。

5次题目的算法均用Java写的，为了简单，输出在控制台上。最后一个医生与病人问题写了一个界面。

* 生产者与消费者 和 读者写者问题都是**两类物品**同时申请**一类资源**。
* 哲学家进餐问题 和 理发师睡觉问题 和 医生看病问题 都是**一类物品**申请**一类或多类资源**。

生产者与消费者问题中，只有缓冲池这一类资源，但是生产者和消费者同时要进行**写操作**（加一或者减一）。

读者和写者问题中，只有作品这一类资源，读者和写者同时要进程**写操作，**但不同的是，写者和写者之间是互斥的！而生产者与生产者之间并不是互斥的。

1. 生产者与消费者问题中，设置每个缓冲池为信号量。当某个生产者访问某个缓冲池，则该缓冲池就会“被锁”。利用Java中的synchronized 关键字。同理，消费者访问某个缓冲池，该缓冲池也同样会被锁。

生产者访问缓冲池有两种情况：

* 所有缓冲池都是满的了：此时生产者则会让权等待，放弃该缓冲池的锁，利用Java的buffer.wait方法
* 仍然有空的缓冲池：找一个空的缓冲池写数据，同时该缓冲池被锁。

同理，消费者也是如此。

1. 读者与写者问题中，设置readerCount和 writerCount变量。当写者写数据的时候，检查是否有读者在读，和是否有写者在写，如果都没有，才会去写数据，写数据的过程中把作品上锁！
2. 哲学家进餐问题中，5个筷子，每个筷子都设置为一个信号量。再设置一个布尔型数组，初始5个FALSE。当一个哲学家准备吃饭时，先检测左右两边的筷子是否都空闲，如果是，则拿起两双筷子，否则等待。吃饭的时候，将左右两双筷子上锁。吃饭之后，同时释放左右两支筷子。
3. 理发师睡觉问题中，只有顾客一类物品申请资源。则设置理发师为信号量。设置isDressing变量，当理发师给顾客理发时，isDressing设置为true。顾客来之后，先检测理发师是否在理发，在理发，则继续检测空闲椅子数量，如果有空闲椅子，则等待，并且在while循环中持续检测理发师是否已经完成了理发。如果没有空闲椅子，则离开。
4. 医生看病问题中，其实和理发师睡觉问题很类似，不过是把一个理发师换成了三个理发师，所以把isDressing变量换成了一个整形变量代表理发师的数量。病人来之后，先检测病椅数量，如果病椅数量为0，则继续检测等椅数量，如果有空闲等椅，则进入循环等待，一直检测是否有病椅可用。如果没有空闲等椅，则离开。

