1. 实验目的：理解多进程、多线程协作时可能存在的问题，学会使用Linux的同步工具

2. 实验目标：

参考课件中的伪代码，用多线程实现读者-写者和哲学家就餐两个经典同步问题。

3. 实验原理：

3.1 哲学家就餐问题可以这样表述，假设有五位哲学家围坐在一张圆形餐桌旁，做以下两件事情之一：吃饭，或者思考。吃东西的时候，他们就停止思考，思考的时候也停止吃东西。餐桌中间有一大碗意大利面，每两个哲学家之间有一只餐叉。因为用一只餐叉很难吃到意大利面，所以假设哲学家必须用两只餐叉吃东西。他们只能使用自己左右手边的那两只餐叉。

在求解该问题时，可以使用一个state数组跟踪每一个哲学家是在进餐，思考，还是饥饿状态。一个哲学家只有在两个邻居都没有进餐时才允许进入进餐状态。因此可以使用一个信号量数组，每一个信号量对应一位哲学家。在所需的叉子被占用时，该哲学家被阻塞。

因此，该问题可以总结为如下流程：

用一个信号量mutex维护一个同一时间只能有一个哲学家拿起或放下叉子的互斥，初始值为1。用一个大小为n的信号量数组s维护同步，初始值全部设为0.

在拿起叉子前，先等待该信号量。获得该信号量后，将该哲学家的state即为hungry，即等待在合适的条件下获得信号量s[i]的状态。

检测左右测哲学家的状态，若左右的哲学家都没有在就餐，即说明可以进入就餐状态，post信号量s[i]。此时哲学家进入就餐状态。可以处理下一个哲学家的拿起放下叉子操作，故post信号量mutex。

若无法获得叉子，即左右的哲学家中有人在进餐，则阻塞当前哲学家，等待获得叉子。

在哲学家就餐结束后，进行放下叉子操作。具体实现需要先获得mutex信号量，保证互斥。

放下叉子，将哲学家i的状态置为thinking。然后检查左右的哲学家有没有被阻塞的，若有则post对应哲学家的信号量，使其可以进餐。

3.2 有读者和写者两组并发进程，共享一个文件，当两个或以上的读进程同时访问共享数据时不会产生副作用，但若某个写进程和其他进程（读进程或写进程）同时访问共享数据时则可能导致数据不一致的错误。因此要求：①允许多个读者可以同时对文件执行读操作；②只允许一个写者往文件中写信息；③任一写者在完成写操作之前不允许其他读者或写者工作；④写者执行写操作前，应让已有的读者和写者全部退出。

若读者需要读数据，需要先获得信号量mutex，对rc进行+1的操作。

在对rc加一后需要判断rc是否为1，即当前读者是不是第一个读者。若是，则需要等待获得信号量db，即需要等待当前没有写者写入时才可以读取。若不是，则继续读，并释放mutex，以处理下一个读者或写者。

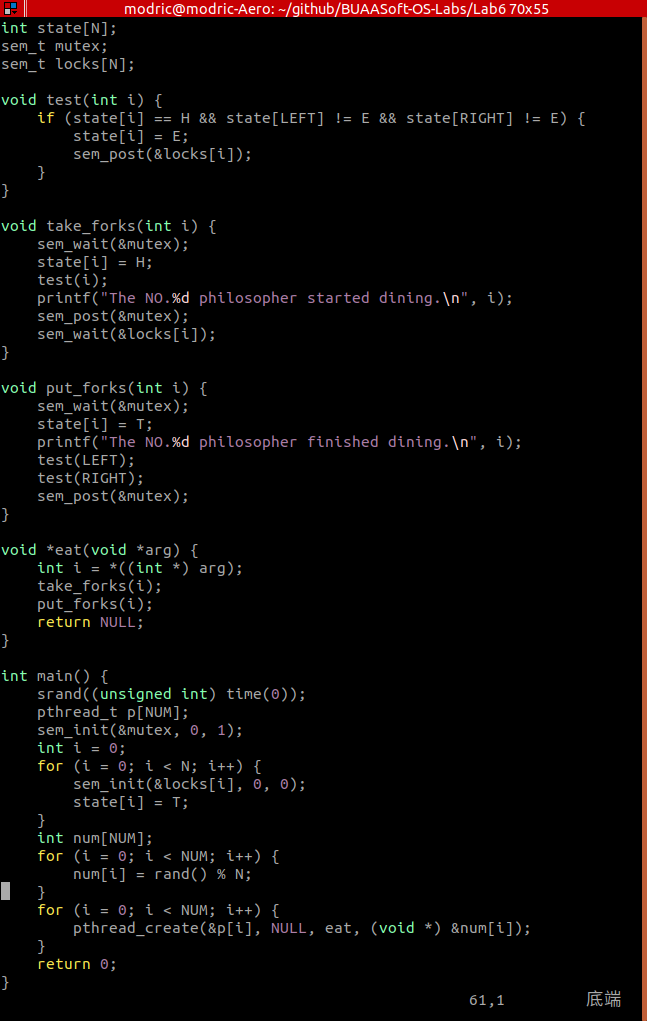
在读者读取完数据后，需要再次等待获得信号量mutex，对rc完成-1的操作。

减一后判断该读者是否为当前最后一个读者，即判断rc是否等于0。若是，则post信号量mutex，以使写者可以对数据进行写入。

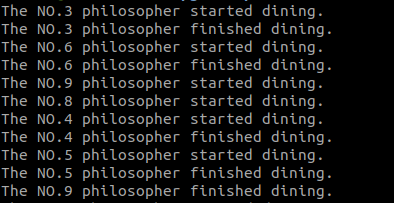
综上，写者在当前没有读者即rc==0，且没有其他写者即可以获得mutex信号量时写入，读者在没有其他写者，即可以获得mutex信号量时可以读入。有效的在保证线程安全的情况下达到了最大的并行度。

4. 实验方案：

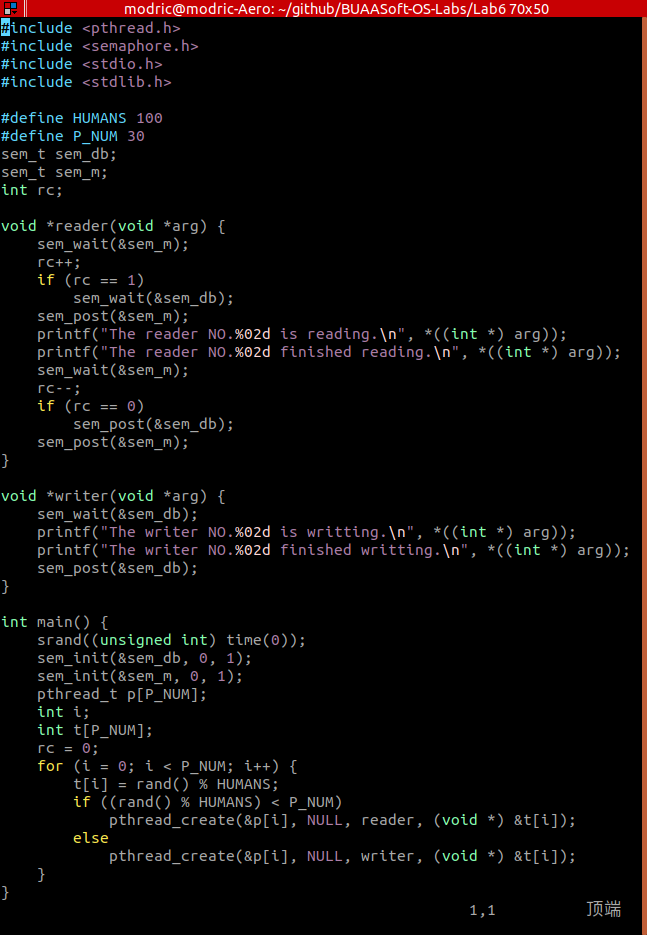
4.1 哲学家就餐问题代码如下：



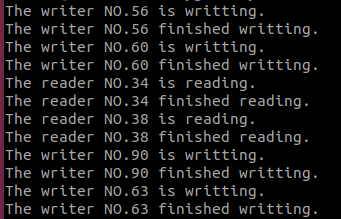
输出（部分）：



读写者问题代码如下：



输出（部分）：



5. 实验结论：

哲学家就餐问题和读者-写者问题是两个经典的多线程同步问题。对这两个问题的研究会让我们更深入的理解并发时遇到的一系列问题及解决策略。