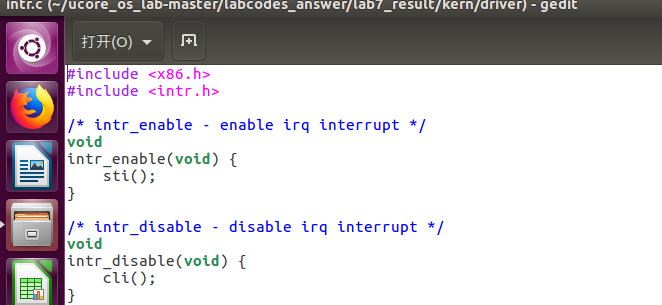
实验5 同步互斥

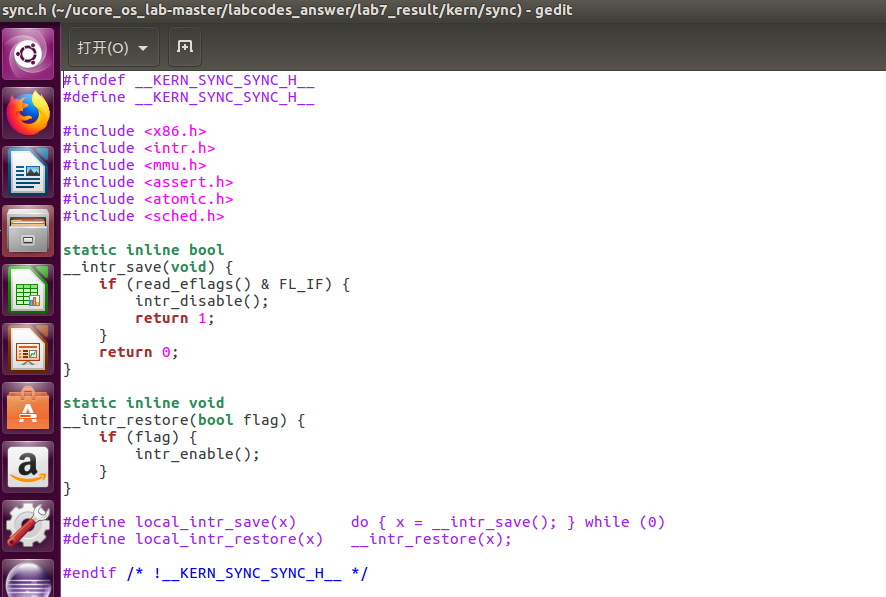
仔细阅读实验文档lab7同步互斥，完成以下练习（不做实验文档中的题目）。扩展练习选做，有能力者完成。

练习1: 了解信号量和管程的实现机制

1. 同步互斥的底层支持是如何实现的？

开关中断

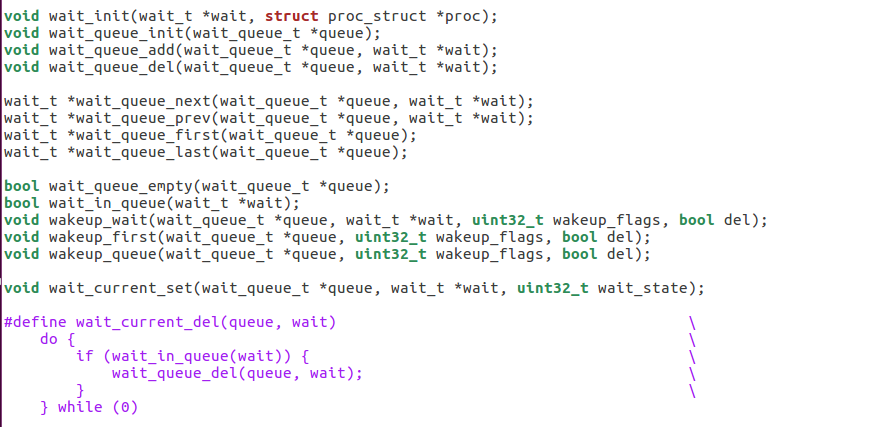




关中断：local\_intr\_save --> \_\_intr\_save --> intr\_disable --> cli

开中断：local\_intr\_restore--> \_\_intr\_restore --> intr\_enable --> sti

等待队列：用户进程或内核线程可以转入休眠状态以等待某个特定事件，当该事件发生时这些进程能够被再次唤醒。内核实现这一功能的一个底层支撑机制就是等待队列（wait queue），等待队列和每一个事件联系起来。需要等待事件的进程在转入休眠状态后插入到等待队列中。当事件发生之后，内核遍历相应等待队列，唤醒休眠的用户进程或内核线程，并设置其状态为就绪状态（runnable state），并将该进程从等待队列中清除。

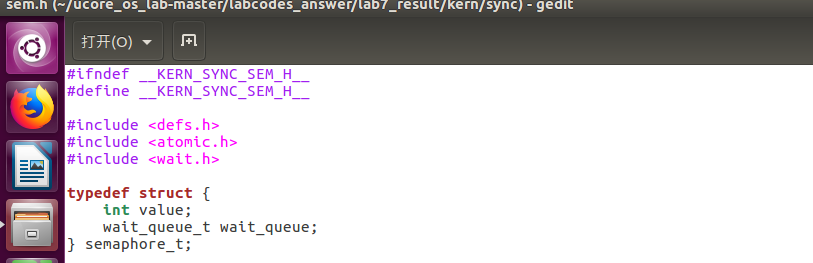


1. 对比原理课上学到的信号量和p，v操作，说明Ucore中信号量机制的实现。

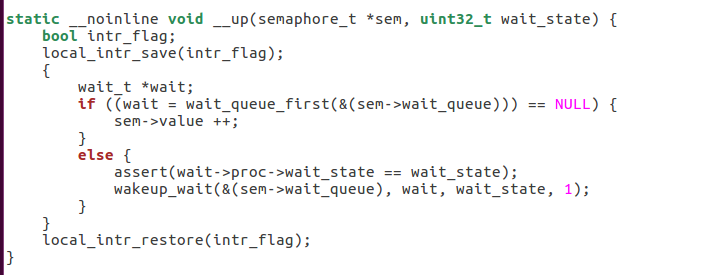
当多个进程可以进行互斥或同步合作时，一个进程会由于无法满足信号量设置的某条件而在某一位置停止，直到它接收到一个特定的信号（表明条件满足了）。为了发信号，需要使用一个称作信号量的特殊变量。为通过信号量s传送信号，信号量的V操作采用进程可执行原语semSignal(s)；为通过信号量s接收信号，信号量的P操作采用进程可执行原语semWait(s)；如果相应的信号仍然没有发送，则进程被阻塞或睡眠，直到发送完为止。

1. Ucore中的信号量是基于信号量和条件变量实现的，请说明其中的数据结构和函数方法的设计。

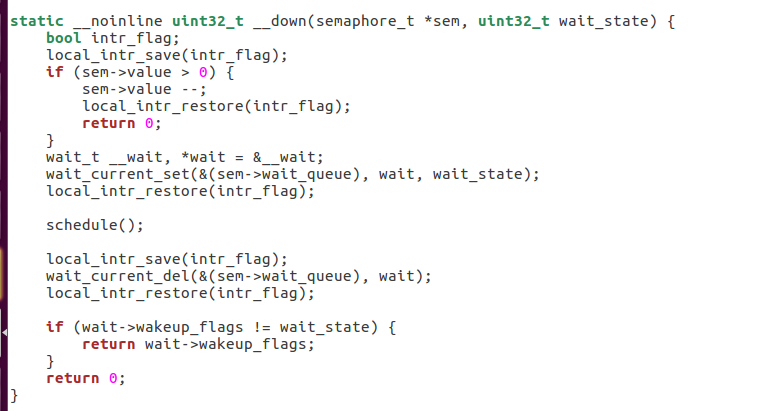
数据结构为：



Up部分：

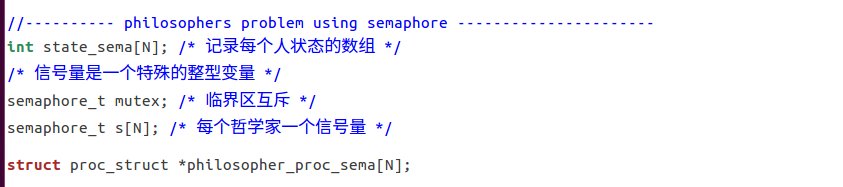


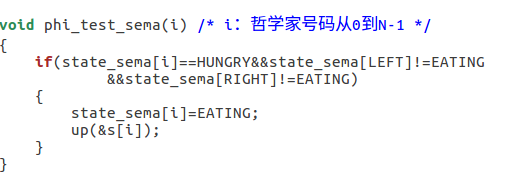
Down部分：

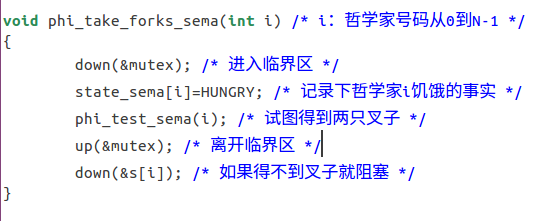


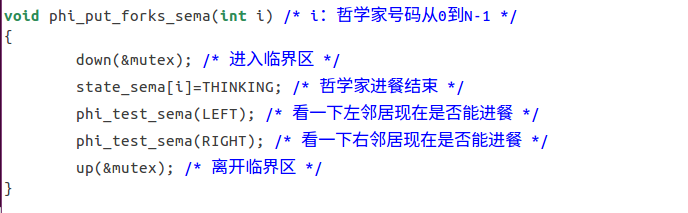
练习2: 了解基于信号量和管程的哲学家就餐问题

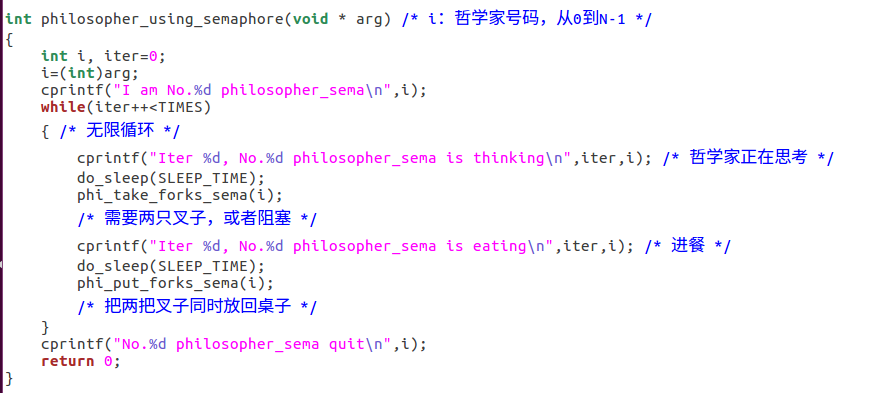
1. 说明ucore中基于信号量的哲学家就餐问题的实现机制。





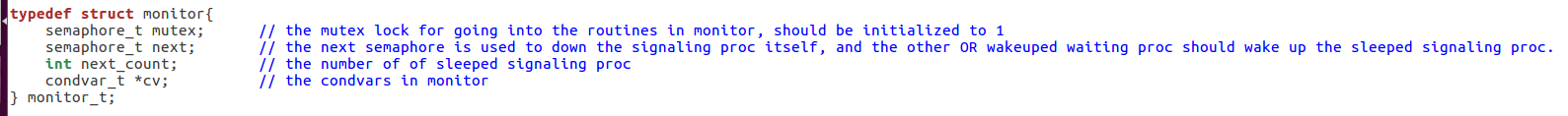


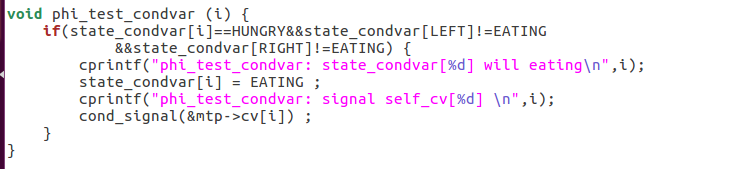




1. 说明ucore中基于管程的哲学家就餐问题的实现机制。

学家释放资源的时候将自己唤醒





扩展练习：了解java中同步互斥的实现机制，说明其与操作系统原理课的管程之间的关系，并用其实现写者优先的读者写者问题。