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？

互斥的底层实现有两方面一方面是硬件支持，一方面是软件设计。包括禁用中断，专用的机器指令，信号量和管程等方法。同步机制主要是@synchronized,NSlock,GCD三种。

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。

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。

具体实现信号量的V操作，首先关中断，如果信号量对应的wait queue中没有进程在等待，直接把信号量的value加一，然后开中断返回；如果有进程在等待且进程等待的原因是semophore设置的，则调用wakeup\_wait函数将waitqueue中等待的第一个wait删除，且把此wait关联的进程唤醒，最后开中断返回。

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。

基于上诉信号量实现可以认为，当多个（>1）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. 说明ucore中基于管程的哲学家就餐问题的实现机制。

管程由四部分组成：管程内部的共享变量，管程内部的条件变量，管程内部并发执行的进程，对局部于管程内部的共享数据设置初始值的语句。管程相当于一个隔离区，它把共享变量和对它进行操作的若干个过程围了起来，所有进程要访问临界资源时，都必须经过管程才能进入，而管程每次只允许一个进程进入管程，从而需要确保进程之间互斥。但在管程中仅仅有互斥操作是不够用的。进程可能需要等待某个条件C为真才能继续执行。在单处理器情况下，将会导致所有其它进程都无法进入临界区使得该条件C为真，该管程的执行将会发生死锁。为此，可引入条件变量（Condition Variables，简称CV）。一个条件变量CV可理解为一个进程的等待队列，队列中的进程正等待某个条件C变为真。每个条件变量关联着一个断言 "断言 (程序)")Pc。当一个进程等待一个条件变量，该进程不算作占用了该管程，因而其它进程可以进入该管程执行，改变管程的状态，通知条件变量CV其关联的断言Pc在当前状态下为真。

①wait\_cv： 被一个进程调用，以等待断言Pc被满足后该进程可恢复执行. 进程挂在该条件变量上等待时，不被认为是占用了管程。

②signal\_cv：被一个进程调用，以指出断言Pc现在为真，从而可以唤醒等待断言Pc被满足的进程继续执行。

有了互斥和信号量支持的管程就可用用了解决各种同步互斥问题。