# 31.sleep和wait的区别(考察的方向是是否会释放锁)#

sleep()方法是Thread类中方法，而wait()方法是Object类中的方法。  
sleep()方法导致了程序暂停执行指定的时间，让出cpu该其他线程，但是他的监控状态依然保持者，当指定的时间到了又会自动恢复运行状态，在调用sleep()方法的过程中，线程不会释放对象锁。而当调用wait()方法的时候，线程会放弃对象锁，进入等待此对象的等待锁定池，只有针对此对象调用notify()方法后本线程才进入对象锁定池准备。

# 32.多线程与死锁#

死锁是指两个或两个以上的进程在执行过程中，因争夺资源而造成的一种互相等待的现象，若无外力作用，它们都将无法推进下去。  
产生死锁的原因：  
一.因为系统资源不足。  
二.进程运行推进的顺序不合适。  
三.资源分配不当。

# 33.如何才能产生死锁#

产生死锁的四个必要条件：  
一.互斥条件：所谓互斥就是进程在某一时间内独占资源。  
二.请求与保持条件：一个进程因请求资源而阻塞时，对已获得的资源保持不放。  
三.不剥夺条件:进程已获得资源，在末使用完之前，不能强行剥夺。  
四.循环等待条件:若干进程之间形成一种头尾相接的循环等待资源关系。

# 34.死锁的预防#

打破产生死锁的四个必要条件中的一个或几个，保证系统不会进入死锁状态。  
一.打破互斥条件。即允许进程同时访问某些资源。但是，有的资源是不允许被同时访问的，像打印机等等，这是由资源本身的属性所决定的。所以，这种办法并无实用价值。  
二.打破不可抢占条件。即允许进程强行从占有者那里夺取某些资源。就是说，当一个进程已占有了某些资源，它又申请新的资源，但不能立即被满足时，它必须释放所占有的全部资源，以后再重新申请。它所释放的资源可以分配给其它进程。这就相当于该进程占有的资源被隐蔽地强占了。这种预防死锁的方法实现起来困难，会降低系统性能。  
三.打破占有且申请条件。可以实行资源预先分配策略。即进程在运行前一次性地向系统申请它所需要的全部资源。如果某个进程所需的全部资源得不到满足，则不分配任何资源，此进程暂不运行。只有当系统能够满足当前进程的全部资源需求时，才一次性地将所申请的资源全部分配给该进程。由于运行的进程已占有了它所需的全部资源，所以不会发生占有资源又申请资源的现象，因此不会发生死锁。  
四.打破循环等待条件，实行资源有序分配策略。采用这种策略，即把资源事先分类编号，按号分配，使进程在申请，占用资源时不会形成环路。所有进程对资源的请求必须严格按资源序号递增的顺序提出。进程占用了小号资源，才能申请大号资源，就不会产生环路，从而预防了死锁。