**多线程**

线程的概念：

线程是操作系统能够进行运算调度的最小单位，它被包含在进程之中，是进程中的实际运作单位

进程：

进程是系统中正在运行的一个程序，进程可以看成程序执行的一个实例。进程是系统资源分配的独立实体，每个进程都拥有独立的地址空间。一个进程无法访问另一个进程的变量和数据结构，

线程安全处理方式：1同步代码块

synchronized (锁对象) {

可能会产生线程安全问题的代码

}

2同步方法

public synchronized void method(){

可能会产生线程安全问题的代码

}

同步方法的对象锁：本类的对象的引用 this

静态同步方法的锁：本类自己.Class（因为静态方法中没有this）

Jdk1.5 lock 替代sychronized

Lock接口作为同步锁：

Lock ck = new ReentrantLock();

Ck.Lock()设置锁 ck.unlock()释放锁

**死锁：**当线程任务中出现了多个同步(多个锁)时，如果同步中嵌套了其他的同步。这时容易引发一种现象：++

那么如何避免和解决死锁问题呢？

1、按顺序加锁

上个例子线程间加锁的顺序各不一致，导致死锁，如果每个线程都按同一个的加锁顺序这样就不会出现死锁。

2、获取锁时限

每个获取锁的时候加上个时限，如果超过某个时间就放弃获取锁之类的。

3、死锁检测

按线程间获取锁的关系检测线程间是否发生死锁，如果发生死锁就执行一定的策略，如终断线程或回滚操作等。

wait（） :等待，将正在执行的线程释放其执行资格 和 执行权，并存储到线程池中。

notify（）：唤醒，唤醒线程池中被wait（）的线程，一次唤醒一个，而且是任意的。

notifyAll（）： 唤醒全部：可以将线程池中的所有wait() 线程都唤醒

这三个方法都是object类，平时都是用锁对象去调用。