```
package com.sxt.others;
/**
 * 可重入锁:锁可以延续使用+计数器
 * @author 江
 *
 */

public class LockTest03 {
	ReLock lock=new ReLock();
	public void a() throws InterruptedException {
		lock.lock();
		System.err.println(lock.getHoldCount());
		doSomething();
		lock.unlock();
		System.err.println(lock.getHoldCount());
	}

	public void doSomething() throws InterruptedException {
		lock.lock();
		System.err.println(lock.getHoldCount());
		lock.unlock();
		System.err.println(lock.getHoldCount());
	}
	public static void main(String[] args) throws InterruptedException {
		LockTest03 lt=new LockTest03();
		lt.a();

		Thread.sleep(1000);

		System.err.println(lt.lock.getHoldCount());
	}

}

class ReLock{
```

```
        //是否占用
    private boolean isLocked=false;
    Thread lockedBy=null;//存储线程
    private int holdCount=0;

    //使用锁
    public synchronized void lock() throws InterruptedException {
            Thread t=Thread.currentThread();
            while(isLocked&&lockedBy!=t) {
                    wait();
            }
            isLocked=true;
            lockedBy=t;
            holdCount++;
    }

    //释放锁
    public synchronized void unlock() {
            if(Thread.currentThread()==lockedBy) {
            holdCount--;
            if(holdCount==0) {
                    isLocked=false;
                    notify();
                    lockedBy=null;
            }

    }
    }

    public int getHoldCount() {
            return holdCount;
    }
}
```