

中山大学数据科学与计算机学院

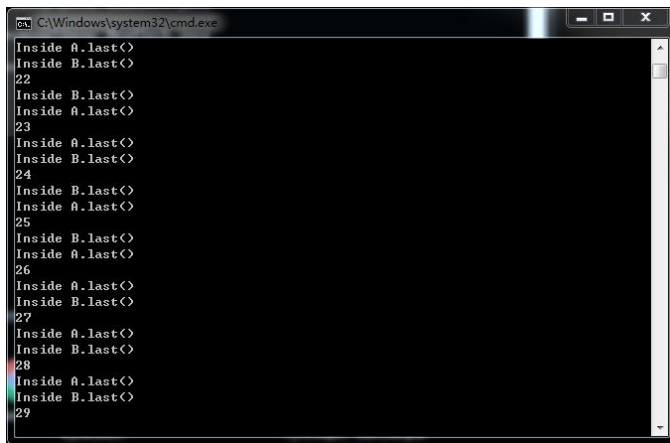
嵌入式实验报告

(2016 学年秋季学期)

教学班级	M2	专业（方向）	移动互联网
学号	14353232	姓名	马发潮

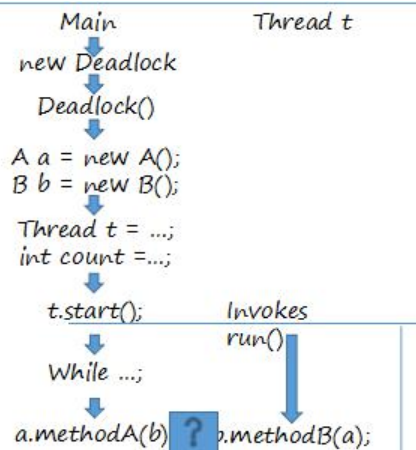
本次的实验是将所给代码跑几遍，分析一下代码和输出结果看到底是为什么发生了死锁和将死锁的几个发生原因列出。

代码跑出的结果之一，count=20000



```
C:\Windows\system32\cmd.exe
Inside A.last()
Inside B.last()
22
Inside B.last()
Inside A.last()
23
Inside A.last()
Inside B.last()
24
Inside B.last()
Inside A.last()
25
Inside B.last()
Inside A.last()
26
Inside A.last()
Inside B.last()
27
Inside A.last()
Inside B.last()
28
Inside A.last()
Inside B.last()
29
```

我们可以看到，输出有 A,B 也有 B,A



```

class Deadlock implements Runnable{

    A a=new A();
    B b=new B();

    Deadlock(){
        Thread t=new Thread(this);
        int count = 20000;

        t.start();
        while(count-->0);
        a.methodA(b);
    }

    public void run(){
        b.methodB(a);
    }
    public static void main(String args[]){
        new Deadlock();
    }
}

```

根据代码和流程图分析，创建了主线程 Deadlock，然后在其中创建了子线程 t，在 count 前 b 寻找 a，在 count 后 a 寻找 b，两个线程同时执行，所以当 count 放在恰当的大小的时候，就能够让 a 寻找 b 和 b 寻找 a 同时执行，此时就形成了死锁。

产生死锁的原因主要是：

- (1) 因为系统资源不足。
- (2) 进程运行推进的顺序不合适。
- (3) 资源分配不当等。

如果系统资源充足，进程的资源请求都能够得到满足，死锁出现的可能性就很低，否则就会因争夺有限的资源而陷入死锁。其次，进程运行推进顺序与速度不同，也可能产生死锁。

产生死锁的四个必要条件：

- (1) 互斥条件：一个资源每次只能被一个进程使用。
- (2) 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。
- (3) 不剥夺条件：进程已获得的资源，在未使用完之前，不能强行剥夺。
- (4) 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。

这四个条件是死锁的必要条件，只要系统发生死锁，这些条件必然成立，而只要上述条件之一不满足，就不会发生死锁。