**中山大学数据科学与计算机学院本科生实验报告**

**（2017学年秋季学期）**

课程名称：嵌入式系统 任课教师：黄凯

|  |  |  |  |
| --- | --- | --- | --- |
| 年级 | **1504** | 专业（方向） | **软件工程（移动信息工程）** |
| 学号 | **15352072** | 姓名 | **党博欣** |
| 电话 | **13609750732** | Email | **2608630476@qq.com** |
| 开始日期 | **2017.9.22** | 完成日期 | **2017.9.23** |

**1.实验目的**

通过实验，结合死锁产生的四个必要条件，分析给出的程序为什么会出现死锁。如果改动数据，结果会有什么变化并分析。

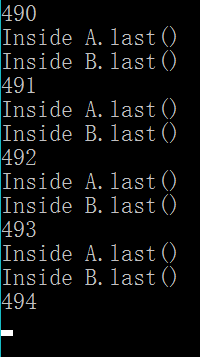
1. **实验过程**

先定义两个原子类class A、class B和一个class Deadlock，将其保存为 Deadlock.java，然后javac Deadlock.java，编译java文件，接着运行批处理文件。

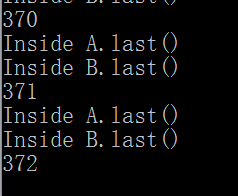
1. **实验结果**

当count为2000时，结果如下：

第一次运行：



第二次运行：

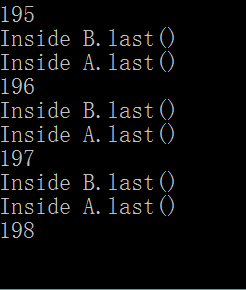


第三次运行：

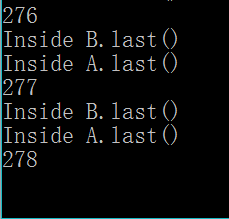
## 

当count为10000:

第一次运行：



第二次运行：



## 第三次运行：

## 

## 实验分析

（1）产生死锁的四个必要条件：

互斥条件：至少有一个资源必须处于非共享模式，即一次只有一个进程使用。

占用并等待：一个进程必须至少占有一个资源，并等待另一资源，而该资源为其他进程所占有。

非抢占:资源不能被抢占，即资源只能在进程完成任务后自动释放。

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

1. 改变count，试了多次，发现每次发生死锁时，count值都不一样，随机出现死锁。

分析程序运行过程：

一共有两个线程：主线程和线程t。

（1）主线程执行：执行main函数，创建Deadlock对象，调用A()、B()，执行Deadlock的构造函数，创建线程t，执行t.start()。

（2）主线程被放入就绪队列，线程t开始执行。当线程t执行时，输出“Inside A.last()”。

（3）当线程t的时间片用完时，线程t被放入就绪队列，主线程继续执行，count--，count为0后，之后输出“Inside B.last()”。

分析：当两个线程同时发生：一个线程在执行原子操作，另一个线程想调用方法时，就会出现死锁。A占用了A的methodA()函数，想要调用B的last()函数，而与此同时，B占用了B的methodB()函数想要A的last()函数，满足了占有且等待的条件、互斥（synchronized实现）、非抢占（Java的两个进程具有相同的优先级）、满足了占有且等待的条件、循环等待（A和B相互等待）。

5.实验总结与心得

虽然上学期已经学过操作系统，但是到现在知识有些遗忘，通过这个实验，我重新巩固了关于死锁发生的条件，也知道了怎么分析产生死锁。