

电子科技大学计算机科学与工程学院

标准实验报告

(实验) 课程名称 计算机操作系统

电子科技大学教务处制表

电子科技大学

实验报告

学生姓名： 闫尔翀

学 号： 2013060202016

指导教师： 薛瑞尼

实验地点： A2-412

实验学时： 2

一、实验项目名称： 银行家算法程序

二、实验内容及要求：

输入

p: 进程数量

r: 资源数量

各进程的 max, allocation

输出

若产生死锁，打印提示：死锁状态。

否则，给出一种调度顺序。

三、实验原理：

1. 安全状态

指系统能按照某种顺序如 $\langle P_1, P_2, \dots, P_n \rangle$ (称为 $\langle P_1, P_2, \dots, P_n \rangle$ 序列为安全序列), 为每个进程分配所需的资源, 直至最大需求, 使得每个进程都能顺利完成。

2. 银行家算法

假设在进程并发执行时进程 i 提出请求 j 类资源 k 个后, 表示为 $Request_i[j]=k$ 。系统按下述步骤进行安全检查:

- (1) 如果 $Request_i \leq Need_i$ 则继续以下检查, 否则显示需求申请超出最大需求值的错误。
- (2) 如果 $Request_i \leq Available$ 则继续以下检查, 否则显示系统无足够资源, P_i 阻塞等待。
- (3) 系统试探着把资源分配给进程 P_i , 并修改下面数据结构中的数值:
 $Available[j] := Available[j] - Request_i[j];$
 $Allocation[i, j] := Allocation[i, j] + Request_i[j];$
 $Need[i, j] := Need[i, j] - Request_i[j];$
- (4) 系统执行安全性算法, 检查此次资源分配后, 系统是否处于安全状态。若安全, 才正

式将资源分配给进程 P_i ，以完成本次分配；否则，将本次的试探分配作废，恢复原来的资源分配状态，让进程 P_i 等待。

3. 安全性算法

(1) 设置两个向量：

① 工作向量 Work：它表示系统可提供给进程继续运行所需的各类资源数目，它含有 m 个元素，在执行安全算法开始时， $Work := Available$ ；

② Finish：它表示系统是否有足够的资源分配给进程，使之运行完成。开始时先做 $Finish[i] := false$ ；当有足够资源分配给进程时，再令 $Finish[i] := true$ 。

(2) 从进程集合中找到一个能满足下述条件的进程：

① $Finish[i] = false$ ；

② $Need[i, j] \leq Work[j]$ ；若找到，执行步骤(3)，否则，执行步骤(4)。

(3) 当进程 P_i 获得资源后，可顺利执行，直至完成，并释放出分配给它的资源，故应执行：

$Work[j] := Work[j] + Allocation[i, j]$ ；

$Finish[i] := true$ ；

go to step 2;

(4) 如果所有进程的 $Finish[i] = true$ 都满足，则表示系统处于安全状态；否则，系统处于不安全状态。

四、实验目的：

- (1) 了解和理解死锁
- (2) 理解利用银行家算法避免死锁的原理

五、实验器材：

Windows 操作系统 PC 一台，Python，VS2013

六、实验结果及数据分析：

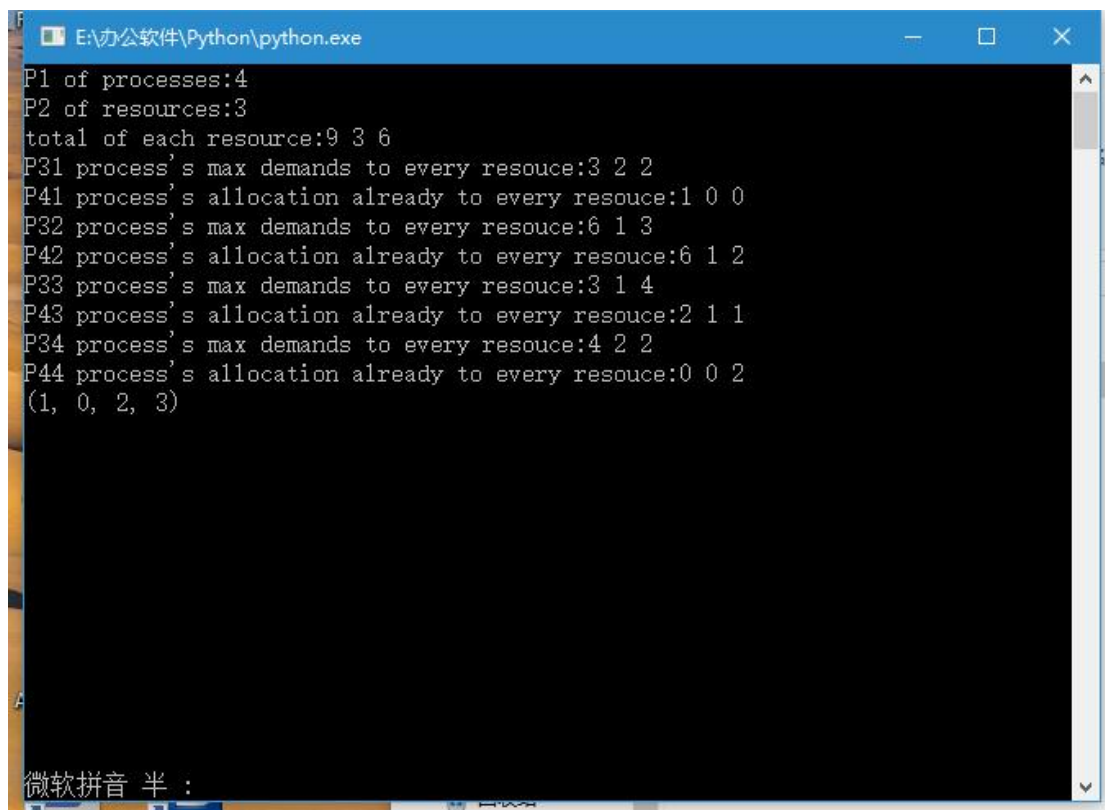
1. 进行正确的验证：

进程数 P1: 4

资源数 P2: 3

资源数量 P3: 9 3 6

各类进程对资源的最大需求及分配进程 allocation 的数量如截图：



```
E:\办公软件\Python\python.exe
P1 of processes:4
P2 of resources:3
total of each resource:9 3 6
P31 process's max demands to every resouce:3 2 2
P41 process's allocation already to every resouce:1 0 0
P32 process's max demands to every resouce:6 1 3
P42 process's allocation already to every resouce:6 1 2
P33 process's max demands to every resouce:3 1 4
P43 process's allocation already to every resouce:2 1 1
P34 process's max demands to every resouce:4 2 2
P44 process's allocation already to every resouce:0 0 2
(1, 0, 2, 3)
```

微软拼音 半 :

输出了正确的运行顺序，没有死锁

2.死锁验证:

进程数 P1: 3

资源数 P2: 2

资源数量 P3: 6 4 5

各类进程对资源的最大需求及分配进程 allocation 的数量如截图:

```
C:\WINDOWS\system32\cmd.exe - lab2.py
C:\Users\yec\Desktop\学习相关\操作系统\实验\OSLAB2>lab2.py
P1 of processes:4
P2 of resources:3
total of each resource:9 8 7
P31 process's max demands to every resource:11 9 3
P41 process's allocation already to every resource:4 5 6
P32 process's max demands to every resource:8 9 6
P42 process's allocation already to every resource:2 5 8
P33 process's max demands to every resource:9 10 10
P43 process's allocation already to every resource:4 3 2
P34 process's max demands to every resource:6 5 8
P44 process's allocation already to every resource:10 2 11
false
```

最后显示为 false，即资源不足，发生死锁。

七、总结及心得体会：

多个进程同时运行时，系统根据各类系统资源的最大需求和各类系统的剩余资源为进程安排安全序列，使得系统能快速且安全的运行进程，不发生死锁。银行家算法是避免死锁的主要方法，它的思路在我们日常的很多方面都值得学习运用。