一、实验内容

死锁避免银行家算法

二、实验目的

多个进程动态地共享系统的资源时可能会产生死锁现象。银行家算法是通过对资源的分配进行动态地检查来达到避免死锁的目的。本实验通过模拟银行家算法的应用，使读者了解银行家算法的执行过程。从而进一步理解死锁产生的条件和避免死锁问题产生的方法。

三、实验要求

1、设资源种类为 N，每类资源的初始可用数目为 1-M，N 和 M 输入或预定义

2、初始时资源全部可用。运行时，随机生成或手工输入新的进程资源请求。

3、编写程序模拟银行家算法

4、显示每次资源请求处理后的处理结果和资源状态。

四、实验分析与设计

1.银行家算法数据结构

1）可利用资源向量 Available

是个含有 m 个元素的数组，其中的每一个元素代表一类可利用的资源数目。如果 Available[j]=K，则表示系统中现有 Rj 类资源 K 个。

2）最大需求矩阵 Max

这是一个 n×m 的矩阵，它定义了系统中 n 个进程中的每一个进程对 m 类资源的最大需求。如果 Max[i,j]=K，则表示进程 i 需要 Rj 类资源的最大数目为 K。

3）分配矩阵 Allocation

这也是一个 n×m 的矩阵，它定义了系统中每一类资源当前已分配给每一进程的资源数。如果 Allocation[i,j]=K，则表示进程 i 当前已分得 Rj 类资源的 数目为 K。

4）需求矩阵 Need。

这也是一个 n×m 的矩阵，用以表示每一个进程尚需的各类资源数。如果 Need[i,j]=K，则表示进程 i 还需要 Rj 类资源 K 个，方能完成其任务。

Need[i,j]=Max[i,j]-Allocation[i,j]

2.算法的实现

（1）初始化

由用户输入数据，分别对可利用资源向量矩阵 AVAILABLE 、 最大需求矩阵 MAX 、分配矩阵 ALLOCATION、 需求矩阵 NEED 赋值。

（2）银行家算法

在避免死锁的方法中，所施加的限制条件较弱，有可能获得令人满意的系统性能。在该方法中把系统的状态分为安全状态和不安全状态，只要能使系统始终

都处于安全状态，便可以避免发生死锁。

银行家算法的基本思想是分配资源之前, 判断系统是否是安全的; 若是, 才分配。它是最具有 代表性的避免死锁的算法。

设进程 cusneed 提出请求 REQUEST [i] ,则银行家算法按如下规则进行判断。

1) 如果 REQUEST [cusneed] [i]<= NEED[cusneed][i] ， 则转(2) ； 否则， 出错。

2) 如果 REQUEST [cusneed] [i]<= AVAILABLE[cusneed][i] ， 则转(3) ； 否则， 出错。

3) 系统试探分配资源，修改相关数据：

AVAILABLE[i]-=REQUEST[cusneed][i];

ALLOCATION[cusneed][i]+=REQUEST[cusneed][i];

NEED[cusneed][i]-=REQUEST[cusneed][i];

4) 系统执行安全性检查，如安全，则分配成立；否则试探险性分配作废，系统恢复 原状， 进程等待。

（3）安全性检查算法

1) 设置两个工作向量 Work=AVAILABLE;FINISH

2) 从进程集合中找到一个满足下述条件的进 程，

FINISH==false;

NEED<=Work;

如找到，执行(3) ； 否则，执行(4)

3) 设进程获得资源，可顺利执行，直至完 成，从而释放资源。

Work+=ALLOCATION;

Finish=true;

GOTO 2

4) 如所有的进程 Finish= true ， 则表 示安全；否则系统不安全。

操作系统安全状态和不安全状态：

安全序列是指一个进程序列{P1， …， Pn}是安全的，如果对于每一个进程 Pi(1≤i≤n)，它以后尚需要的资源量不超过系统当前剩余资源量与所有进程 Pj (j < i) 当前占有资源量之和。

如果存在一个由系统中所有进程构成的安全序列 P1，…，Pn，则系统处于安全状态。安全状态一定是没有死锁发生。

不存在一个安全序列。不安全状态不一定导致死锁。

五、实验运行结果和相关源代码

1.实验结果

初始化时，可选择自动初始化和手动初始化，自动初始化按照课本上的例子初始化，为方便起见我们采用自动初始化，紧接着打印出各种矩阵：

```
输入1选择手动初始化
输入2选择自动初始化
2
最大需求矩阵:
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3
当前已分配矩阵:
0 1 0
2 0 0
3 0 2
2 1 1
0 0 2


剩余需求矩阵:
7 4 3
1 2 2
6 0 0
0 1 1
4 3 1
```

剩余资源：
3 3 2

当前存在安全序列：P1 P3 P0 P2 P4

请输入请求资源的进程号：

可以看出当前存在安全序列，所以可以允许请求资源。
接着我们输入资源请求：

请输入请求资源的进程号：
1

请依次输入所请求资源数量
1 0 2

经检测存在安全序列，允许分配：

存在安全序列P1 P3 P0 P2 P4 ，允许分配

分配后各矩阵情况：

分配后处理结果和资源状态如下：

最大需求矩阵：
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3
当前已分配矩阵：
0 1 0
3 0 2
3 0 2
2 1 1
0 0 2

剩余需求矩阵:
7 4 3
0 2 0
6 0 0
0 1 1
4 3 1

剩余资源:
2 3 0

接着继续申请资源:
由于没有足够资源，不予以分配

请输入请求资源的进程号:
4

请依次输入所请求资源数量
3 3 0
请求失败！ 当前尚无足够资源！

0 号进程请求分配资源，但不存在安全序列可能导致死锁，不予以分配:

请输入请求资源的进程号:
0

请依次输入所请求资源数量
0 2 0
不存在安全序列，不允许分配！！

改成 0 1 0 后允许分配:

请依次输入所请求资源数量
0 1 0
存在安全序列P1 P3 P0 P2 P4 ，允许分配