操作系统 -- 实验四实验报告

实验四：银行家算法的模拟与实现

一、实验题目

> 本实验的内容是要通过编写和调试一个模拟系统动态分配资源的银行家算法程序，有效地避免 死锁发生。 具体要求如下： （1） 初始化时让系统拥有一定的资源； （2） 用键盘输入的方式允许进程动态申请资源； （3） 如果试探分配后系统处于安全状态，则修改系统的资源分配情况，正式分配资源； （4） 如果试探分配后系统处于不安全状态，则提示不能满足请求，恢复原状态并阻塞该进程。

二、实验目的

> (1) 进一步了解进程的并发执行。 (2) 加强对进程死锁的理解，理解安全状态与不安全状态的概念。 (3) 掌握使用银行家算法避免死锁问题。

三、总体设计

> 当一进程提出资源申请时，银行家算法执行下列步骤以决定是否向其分配资源： 1）检查该进程所需要的资源是否已超过它所宣布的最大值。 2）检查系统当前是否有足够资源满足该进程的请求。 3）系统试探着将资源分配给该进程，得到一个新状态。 4）执行安全性算法，若该新状态是安全的，则分配完成；若新状态是不安全的，则恢复原状 态，阻塞该进程。

四、详细设计

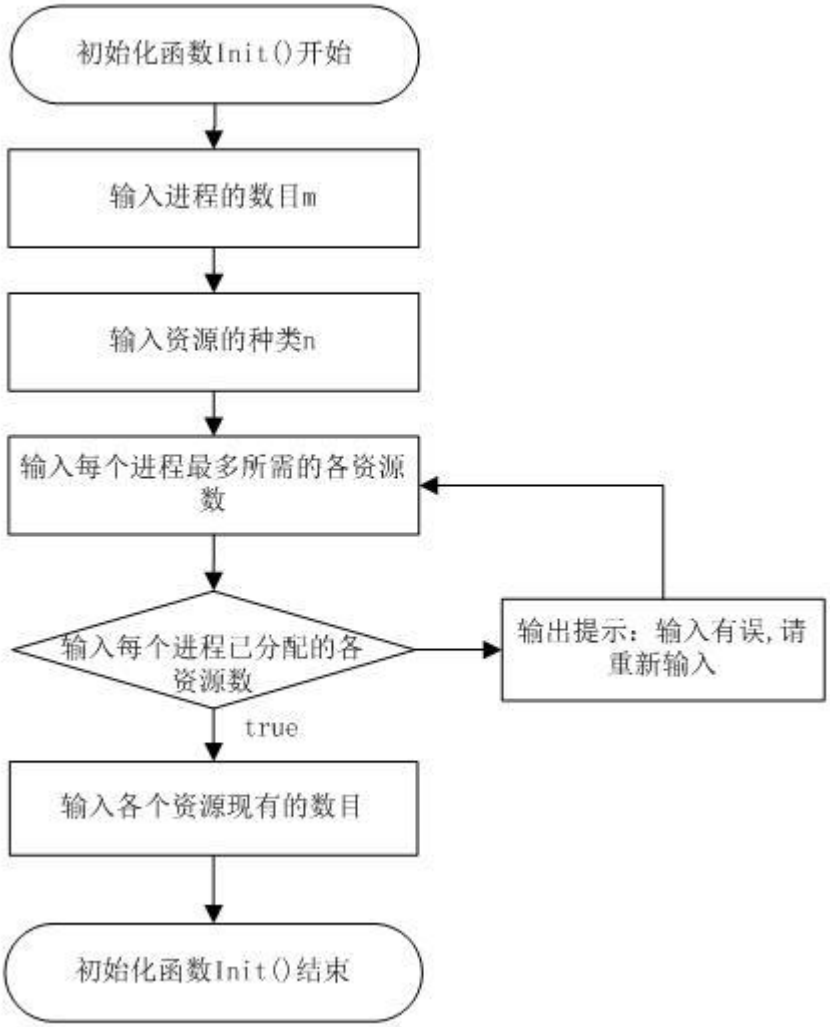
1. 数据结构：

> 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]

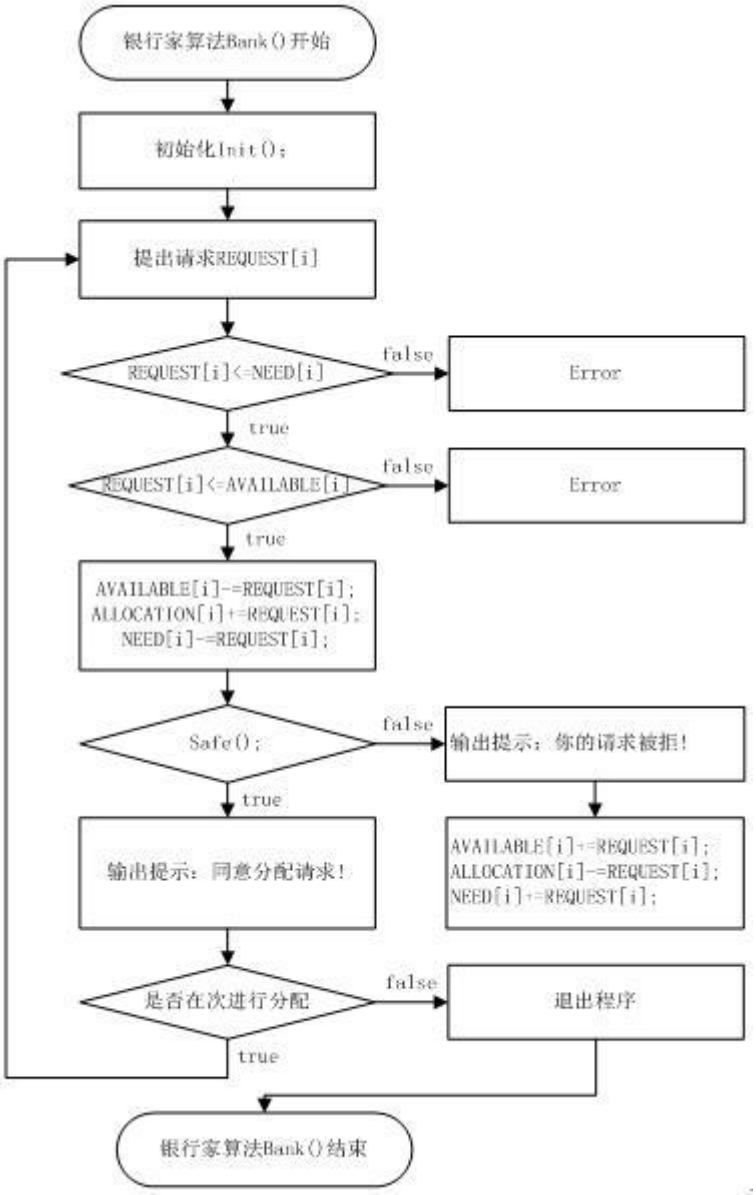
2. 算法流程图：

> 当一进程提出资源申请时，银行家算法执行下列步骤以决定是否向其分配资源： 1）检查该进程所需要的资源是否已超过它所宣布的最大值。 2）检查系统当前是否有足够资源满足该进程的请求。 3）系统试探着将资源分配给该进程，得到一个新状态。 4）执行安全性算法，若该新状态是安全的，则分配完成；若新状态是不安全的，则恢复原状 态，阻塞该进程。

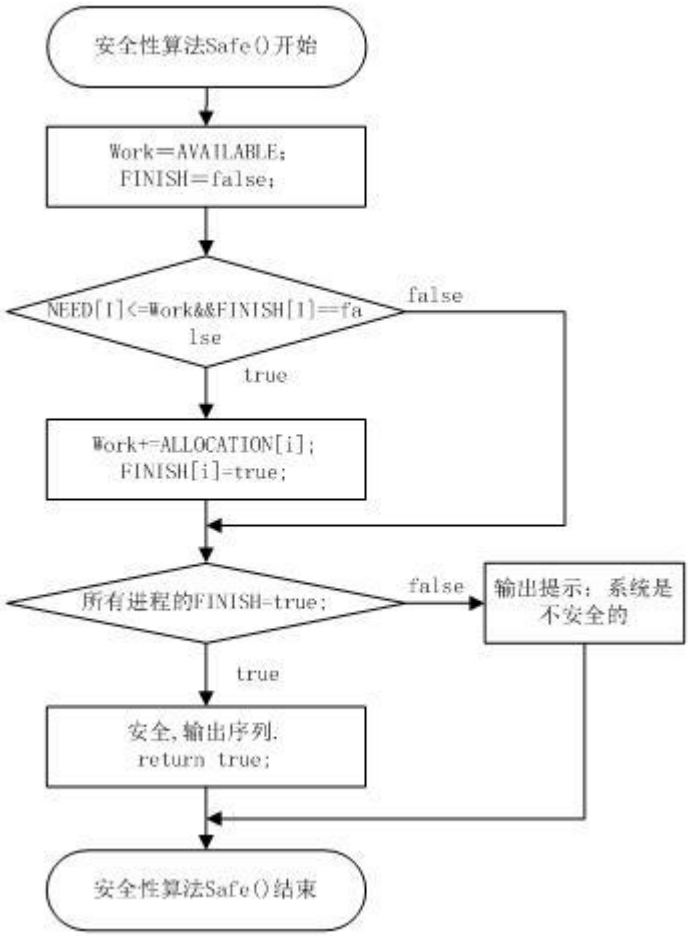
Init 初始化算法流程图：



Bank 银行家算法流程图：

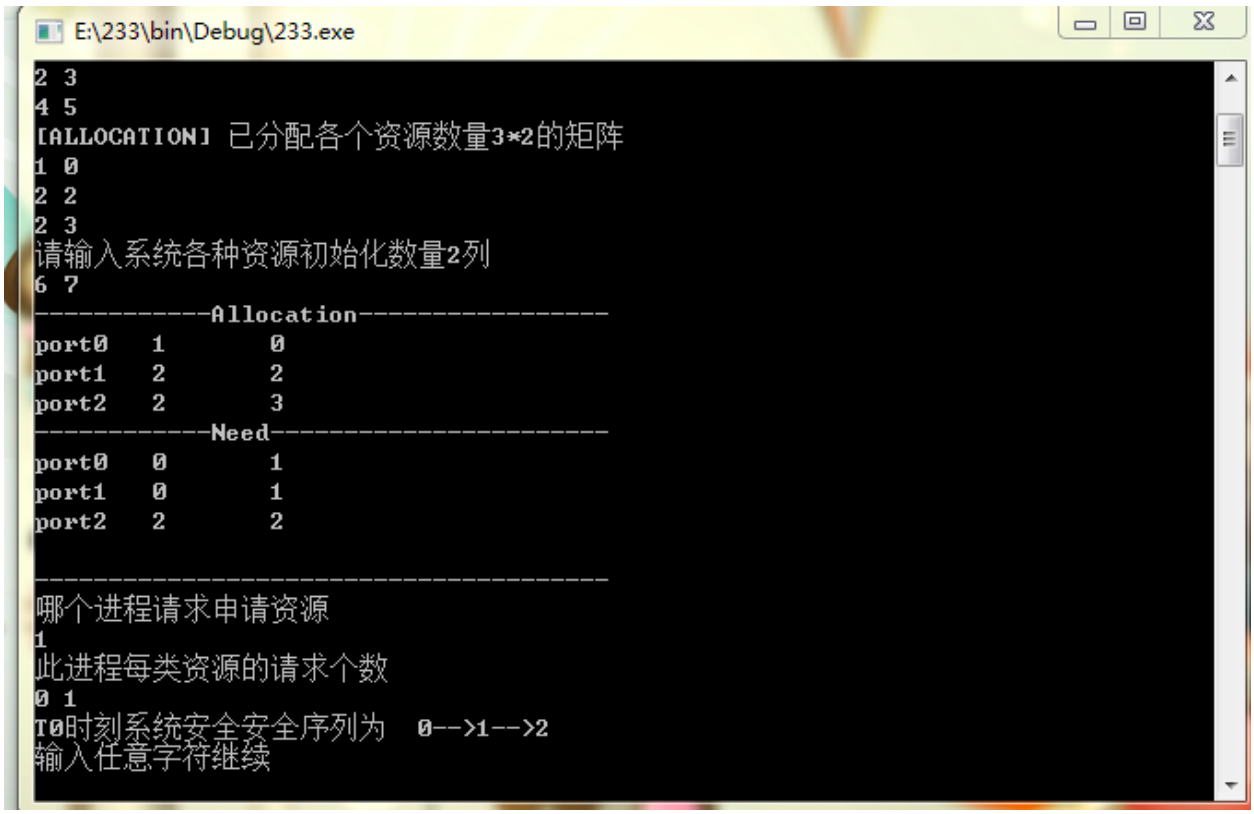


Safe 安全性算法流程图：



五、实验结果与分析

1. 程序运行结果：



2. 实验结果分析：

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。

六、小结与心得体会

> **1.1 银行家算法的实现思想**
>
> 允许进程动态地申请资源，系统在每次实施资源分配之前，先计算资源分配的安全性，若此次资源分配安全（即资源分配后，系统能按某种顺序来为每个进程分配其所需的资源，直至最大需求，使每个进程都可以顺利地完成），便将资源分配给进程，否则不分配资源，让进程等待。
>
> **1.2 死锁的概念**
>
> 死锁是指两个或两个以上的进程在执行过程中，由于竞争资源或者由于彼此通信而造成的一种阻塞的现象，若无外力作用，它们都将无法推进下去。此时称系统处于死锁状态或系统产生了死锁，这些永远在互相等待的进程称为死锁进程。
>
> 银行家算法是避免死锁的一种重要方法。 操作系统按照银行家制定的规则为进程分配资源，当进程首次申请资源时，要测试该进程对资源的最大需求量，如果系统现存的资源可以满足它的最大需求量则按当前的申请量分配资源，否则就推迟分配。当进程在执行中继续申请资源时，先测试该进程已占用的资源数与本次申请的资源数之和是否超过了该进程对资源的最大需求量。若超过则拒绝分配资源，若没有超过则再测试系统现存的资源能否满足该进程尚需的最大资源量，若能满足则按当前的申请量分配资源，否则也要推迟分配。
>
> **1.3 产生死锁的必要条件**
>
> ① 互斥条件：指进程对所分配到的资源进行排它性使用，即在一段时间内某资源只由一个进程占用。如果此时还有其它进程请求资源，则请求者只能等待，直至占有资源的进程用毕释放。
>
> ② 请求和保持条件：指进程已经保持至少一个资源，但又提出了新的资源请求，而该资源已被其它进程占有，此时请求进程阻塞，但又对自己已获得的其它资源保持不放。

> ③ 不剥夺条件：指进程已获得的资源，在未使用完之前，不能被剥夺，只能在使用完时由自己释放。
>
> ④ 环路等待条件：指在发生死锁时，必然存在一个进程——资源的环形链，即进程集合{P0，P1，P2，…，Pn}中的P0正在等待一个P1占用的资源；P1正在等待P2占用的资源，……，Pn正在等待已被P0占用的资源。