**武汉大学计算机学院**

**本科生实验报告**

**模拟银行家算法**

专 业 名 称 ：计算机科学与技术

课 程 名 称 ：操作系统设计

指 导 教 师 ：宋伟

学 生 学 号 ：

学 生 姓 名 ：

二○二一年七月

**郑 重 声 明**

本人呈交的实验报告，是在指导老师的指导下，独立进行实验工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本实验报告不包含他人享有著作权的内容。对本实验报告做出贡献的其他个人和集体，均已在文中以明确的方式标明。本实验报告的知识产权归属于培养单位。

本人签名： 日期： 2021.7.16

摘 要

模拟银行家算法实验的实验目的是加强对资源分配以及如何避免死锁的理解。

实验设计主要遵循实验设计原理。

实验内容主要包括：参照课本上的银行家算法的实例，设计、实现银行家算法。

实验结论为该实验让我重温了银行家算法，加深了我对资源分配和死锁的理解。

**关键词：**银行家算法

**目 录**

**1** **模拟银行家算法**

1.1 实验内容及上机实验所用平台 ……………………………………………………… 1

1.2 数据结构及代码段分析 ……………………………………………………………… 1

1.3 调试过程 ………………………………………………………………………………… 3

1.4 实验总结 ………………………………………………………………………………… 4

**参考文献** …………………………………………………………………………………… 5

**1** **模拟银行家算法**

**1.1 实验内容及上机实验所用平台**

本实验设计、实现银行家算法。

**1.1.1 实验内容**

参照课本上的银行家算法的实例，设计、实现银行家算法。

**1.1.2 实验设计**

（1）假定有M个进程和N种资源，首先给定了最大需求矩阵Max[M][N]、已分配矩阵Allocation[10][10]、需求矩阵Need[10][10]和剩余矩阵Available[10]。

（2）运用银行家算法，反复遍历进程序列，直至找到一个进程的需求矩阵小于等于当前的剩余矩阵，则可使该进程运行并回收其已分配的资源，重复这个步骤直至所有进程都能得到足够的资源或出现死锁。

（3）对用户的输入进行分析，将用户需求先予以满足，然后检查满足后是否仍处于安全状态，如果是则可以进行分配，如果不是则不能分配。

**1.1.3 上机实验所用平台**

开发软件采用VSCode，编程语言使用C++。

**1.2 数据结构及代码段分析**

**1.2.1 数据结构**

定义最大需求矩阵Max[M][N]、已分配矩阵Allocation[10][10]、需求矩阵Need[10][10]和剩余矩阵Available[10]。Max[i][j]表示第i号进程的第j号资源的最大需求量；Allocation[i][j]表示第i号进程的第j号资源的已分配资源数目；Need[10][10]表示第i号进程的第j号资源的需求资源数目；Available[i]表示第i号资源的剩余量。

**1.2.2 代码段分析**

重点代码段如下：

bool safe(){

bool done[10];

int temp[10];

memset(done, false, sizeof(done));

memcpy(temp, Available, sizeof(Available));

bool flag1 = true;

while(flag1){

flag1 = false;

for(int i = 0; i < m; i++){

if(!done[i]){

bool flag2 = true;

for(int j = 0; j < n; j++){

if(temp[j] < Need[i][j]){

flag2 = false;

break;

}

}

if(flag2){

done[i] = true;

flag1 = true;

for(int j = 0; j < n; j++)

temp[j] += Allocation[i][j];

break;

}

}

}

}

for(int i = 0; i < m; i++)

if(!done[i])

return false;

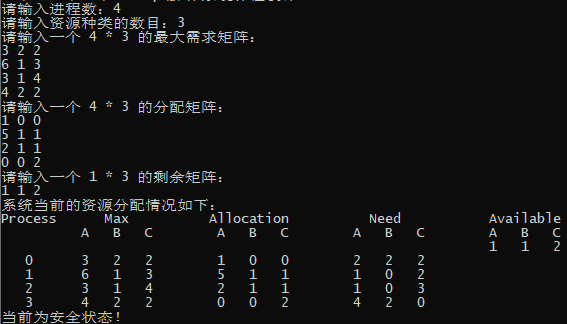
return true;

}

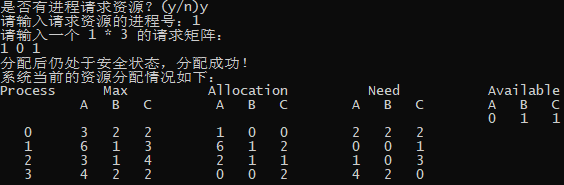
这是检查是否处于安全状态的函数，done数组表示进程是否已经得到足够资源，temp数组是Available数组的一个副本，在检查过程中只对temp数组进行操作而不改变实际的Available数组，代码的循环中遍历了各个进程，如果某个进程没有得到足够资源并且它的Need数组小于等于temp数组，则可以为该进程分配资源，该进程得到满足，回收其资源时要将temp数组加上该进程的Allocation数组。重复遍历直至所有进程都得到满足或者形成死锁，即可返回当前的状态信息。

**1.3 调试过程**

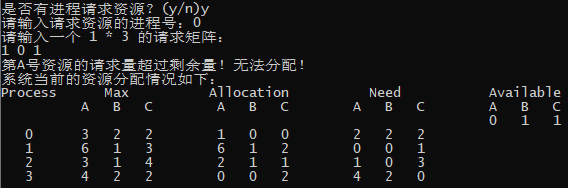
我们按照课本上的示例来调试程序，有4个进程和3种资源，具体调试过程如下：



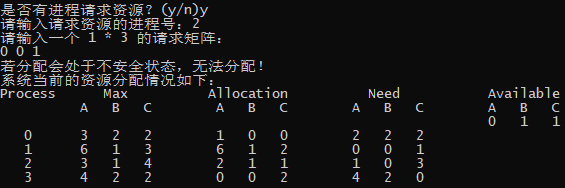
图**1.1** 初始状态



图**1.2** 进程1请求资源(A, B, C)=(1, 0, 1)



图**1.3** 进程0请求资源(A, B, C)=(1, 0, 1)



图**1.4** 进程2请求资源(A, B, C)=(0, 0, 1)

**1.4 实验总结**

该实验让我重温了银行家算法，加深了我对资源分配和死锁的理解。在编写代码的过程中我也遇到了一些问题，例如在检测是否为安全状态时直接改动了Available数组导致无法还原为原来的状态；在输出资源分配表时在格式上也遇到了一些问题，花费了一些时间。但这些问题在经过我的仔细检查和更加谨慎的思考下都得到了有效的解决。总体来说我认为该实验原理并不是特别复杂，但能够加强我对银行家算法的运用和对资源分配的认知，让我有许多收获！

**参考文献**

[1] 郑鹏，曾平，金晶.计算机操作系统[M].武汉:武汉大学出版社，2014.7

教师评语评分

评语：

评分：

评阅人：

年 月 日

（备注：对该实验报告给予优点和不足的评价，并给出百分之评分。）