**武汉大学计算机学院**

**本科生实验报告**

**银行家算法**

专 业 名 称 ：计算机科学与技术

课 程 名 称 ：操作系统课程设计

指 导 教 师 ：宋伟

学 生 学 号 ：2019302070035

学 生 姓 名 ：刘涛榕

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：刘涛榕 日期：2021.7.20

**目 录**

[**银行家算法** 4](#_Toc77961205)

[**一、 实习内容及上机实验所用平台** 4](#_Toc77961206)

[**二、 数据结构及代码段分析** 4](#_Toc77961207)

[**三、 调试过程** 9](#_Toc77961208)

[**四、 实验总结** 13](#_Toc77961209)

**银行家算法**

**一、 实习内容及上机实验所用平台**

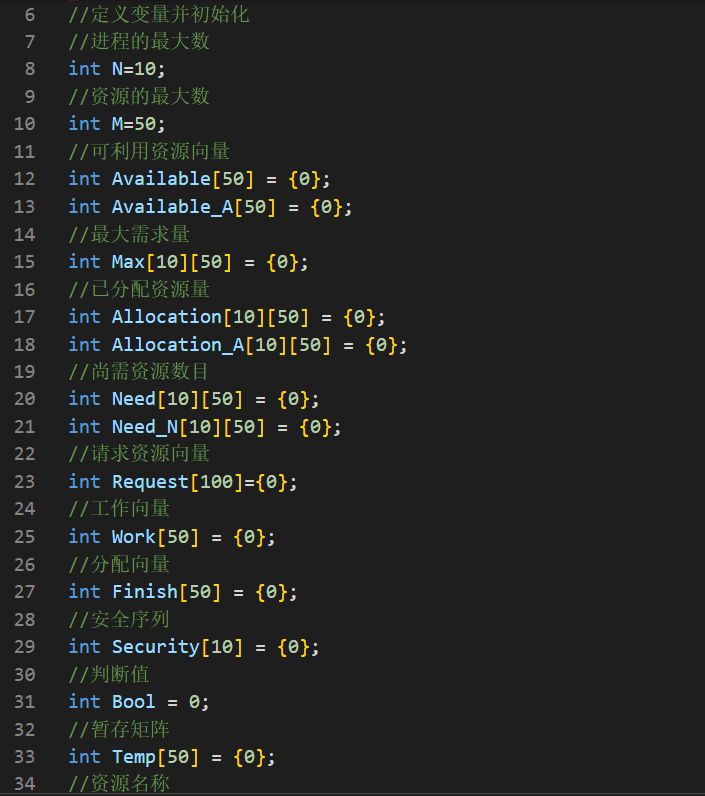
请参照课本上的银行家算法的实例，设计、实现银行家算法。

输入：某个进程请求的资源数量；

输出：已分配或者拒绝本次请求。

要求循环显示提示信息“请输入请求资源的数量和进程号：”。

上机实验所用平台：VS Code

**二、 数据结构及代码段分析**

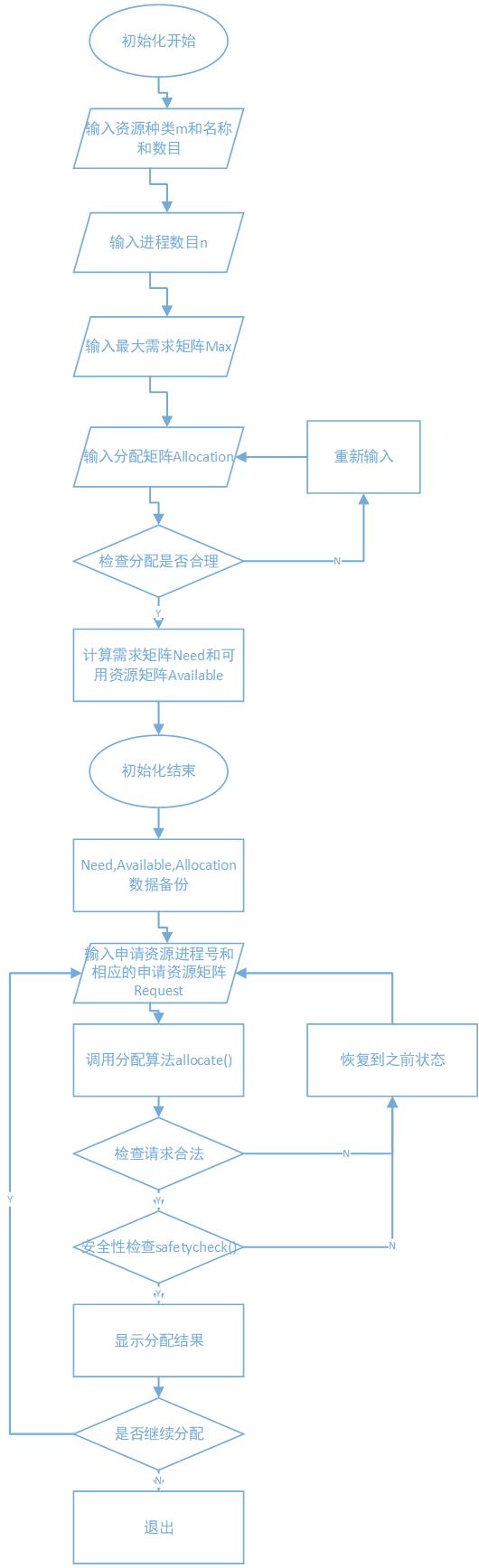
可利用资源向量Availble：含有m个元素的数组，其中的每一个元素代表一类可用的资源数目。Available[j]=k，则表示系统中现有第j类资源K个。这里设置为最大量为50

最大需求矩阵（MAX）：为n×m矩阵，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。Max[i,j]=K，则表示进程i需要第j类资源最大数目为K个。

分配矩阵（Allocation）：为n×m矩阵，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。Allocation[i,j]=K，则表示进程i当前已分配的第j类资源的数目为K。

需求矩阵（Need）：为n×m矩阵，表示每个进程尚需的各类资源数，Need[i,j]=K，则表示进程i还需要第j类资源数目为K个。

**后三部分代码太长不方便截图，这里展示算法的流程图**

**银行家算法（整个函数的流程）**

设Request i是进程Pi的请求矢量，如果Request i[j]=K，表示进程Pi需要第j类资源K个。当Pi发出资源请求后，系统按下述步骤进行检测：

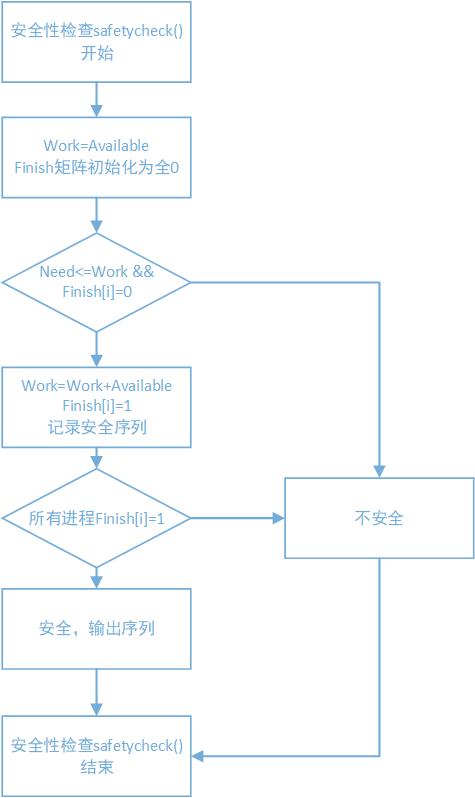
1. 如果Request i[j] ≤Need[i,j]，便转向步骤2，否则认为出错，因为它所需的资源数超过了它所宣布的最大值。
2. 如果Request i[j] ≤Available[i,j]，便转向步骤3，否则，表示尚无足够资源，Pi须等待
3. 系统试探着把资源分配给进程Pi，并修改下面数据结构中的数值：

Available[j]=Availabe[j]-Request i[j]

Allocation[i,j]=Allocation[i,j]+Request i[j]

Need[i,j]=Need[i,j]-Request i[j]

1. 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Pi，以完成本次分配

**安全性检查算法（safetycheck函数流程）**

1. 设置两个矩阵。

工作矩阵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；

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Finish[i]=false；当有足够资源分配给进程Pi时，再令Finish[i]=true;

1. 从进程集合中找到一个能满足下述条件的进程：

Finish[i]=false;

Need[i,j]=Work[j];

若找到，执行下一步骤，否则，执行步骤4。

1. 当进程Pi获得资源后，可顺利执行，直至完成，并释放分配给它的资源

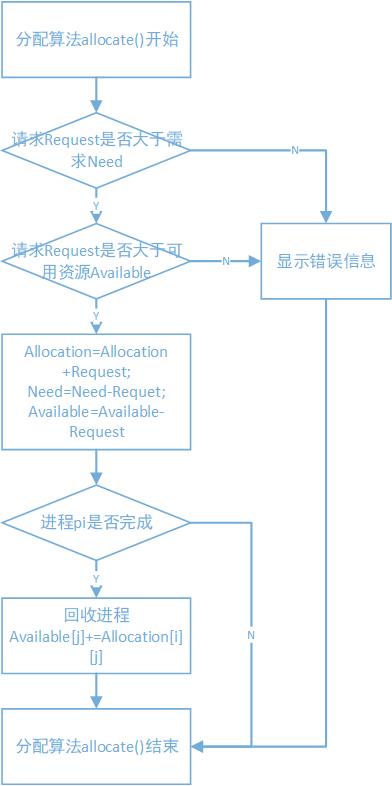
Work[j]=Work[j]+Allocation[i,j];

Finish[i]=1;

执行步骤2

1. 如果所有进程的Finish[i]=1满足，则表示系统处于安全状态；否则系统处于不安全状态

**分配函数（allocate）**



Available[j]=Availabe[j]-Request i[j]

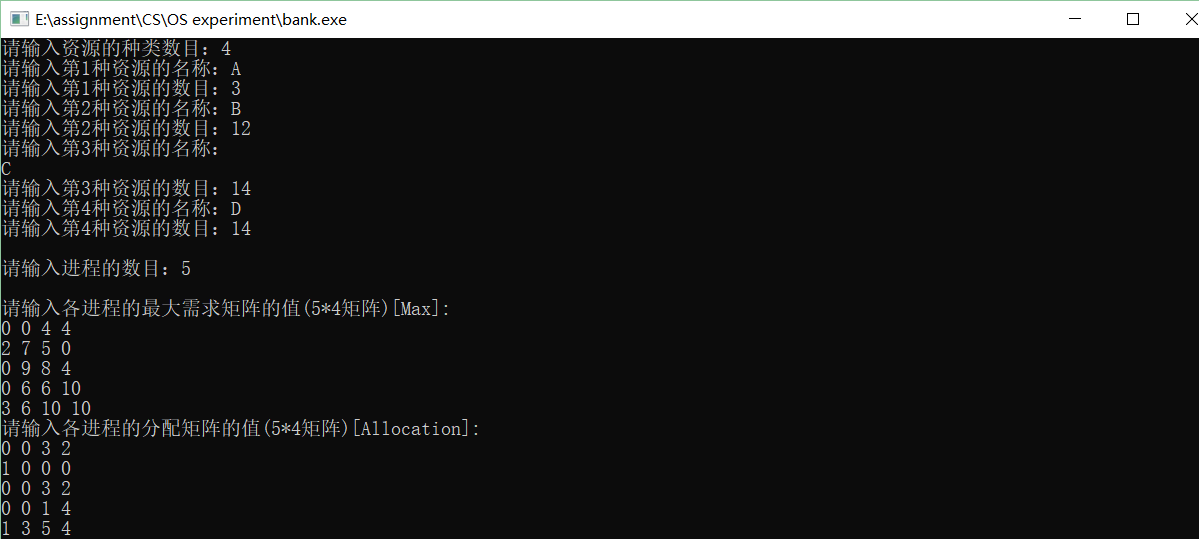
Allocation[i,j]=Allocation[i,j]+Request i[j]

Need[i,j]=Need[i,j]-Request i[j]

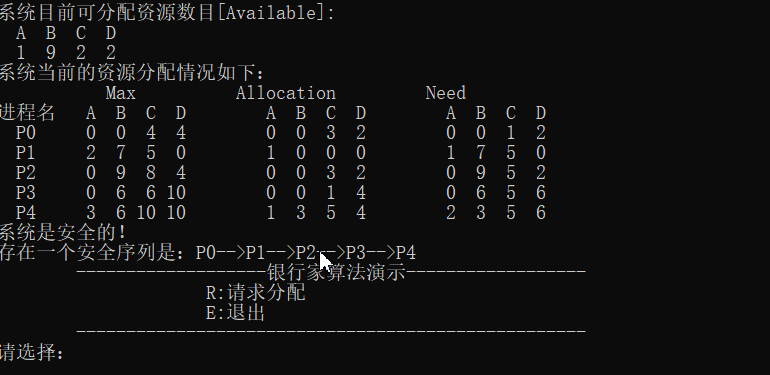
**三、 调试过程**

**测试**

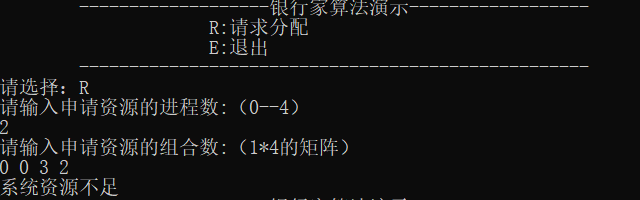
输入资源种类及其名称数量，进程数量，最大需求矩阵和分配矩阵



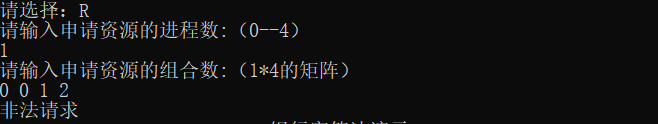
得到可分配矩阵和安全序列检查结果：



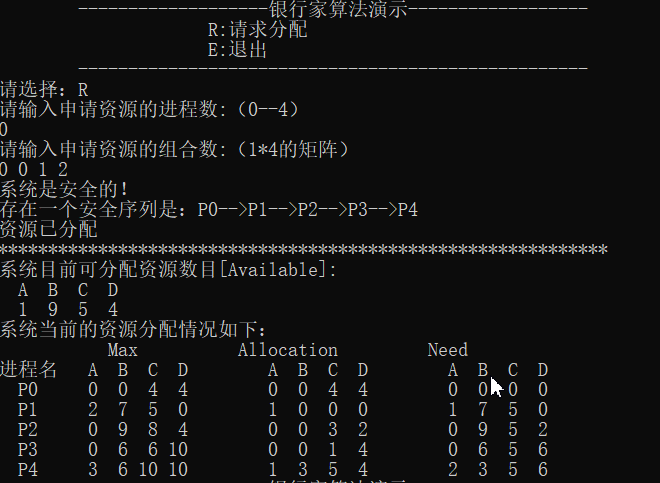
输入R请求分配：可看出请求资源大于系统已有资源会报错



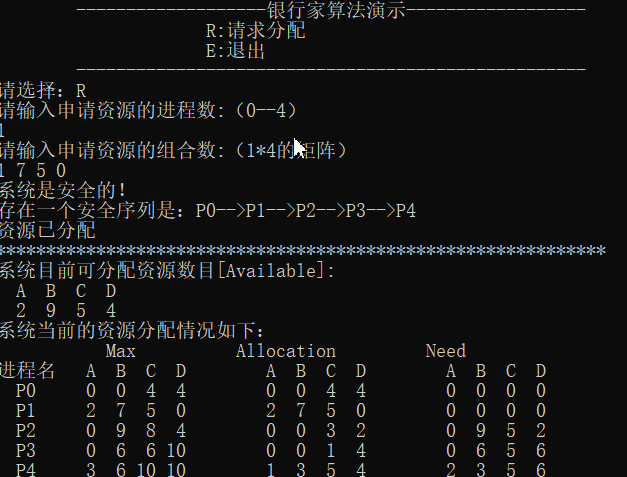
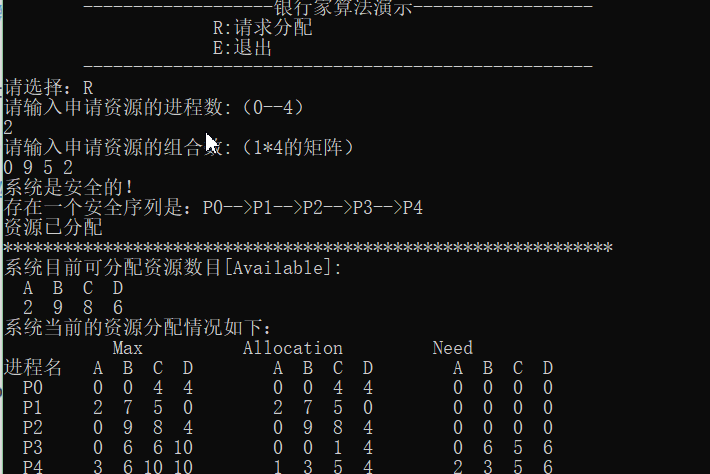
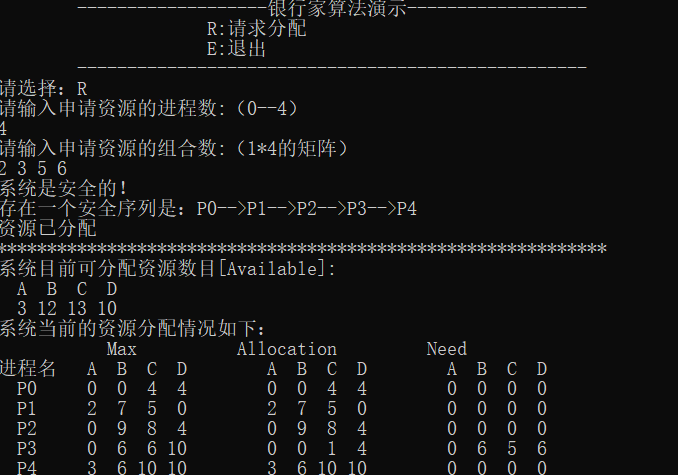
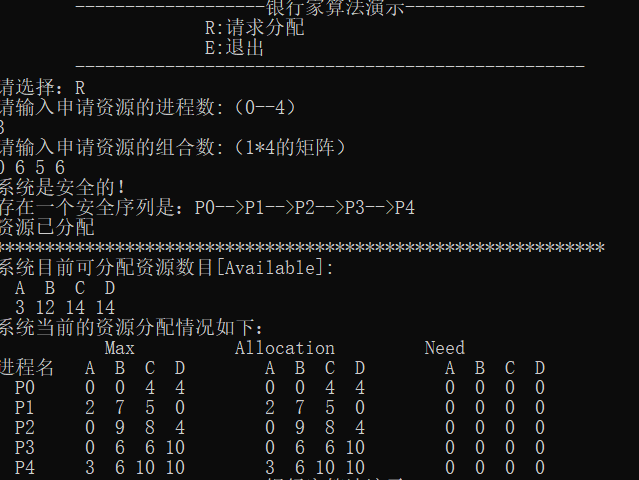
输入R请求分配：可看出请求资源大于需求会报错：



输入R请求分配：请求成功



再继续请求分配：

最后分配完毕和，可看出和之前的资源总数保持一致

**四、 实验总结**

首先重新熟悉和复习了银行家算法，对预防死锁有更深刻的认识。

其次了解死锁产生的条件和原因，并采用银行家算法有效得防止死锁的发生，以加深对课堂上所授知识的理解。

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。）