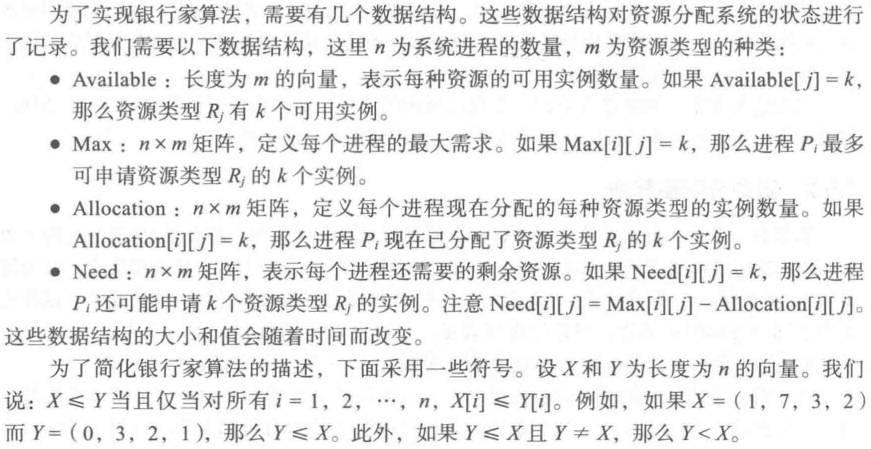
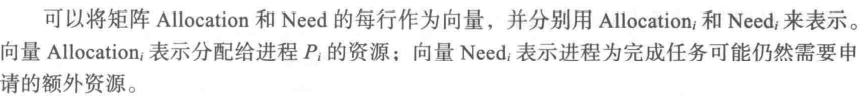
**Project 6: Banker's Algorithm**

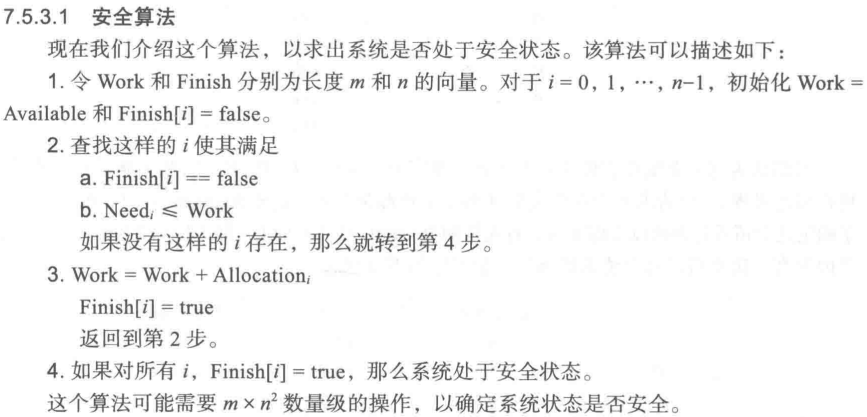
课本中提供了两种可供选择的语言——C或者java。我选择使用C语言来完成该项目。

（一）问题分析

需要实现银行家算法。









（二）实现细节

一、全局变量

1、available[]: the available amount of each resource

2、maximum[][]: the maximum demand of each customer

3、allocation[][]: the amount currently allocated to each customer

4、need[][]: the remaining need of each customer

5、request[]: 记录RQ指令请求的资源

6、release[]: 记录RL指令想要释放的资源

二、函数

1、request\_resource()

（1）根据custom\_num和request[]计算新状态的new\_available[]、new\_allocation[][]、new\_need[][].计算过程中可做简单判断：如果request > available或者new\_allocation > maximum，那么立刻可以判断出该请求不能被批准。

（2）判断新状态是否为安全状态。用new\_available[]初始化work[]，用finish[]记录进程的完成状态。遍历寻找下一个可以完成的进程，即未完成且need < work的进程。若找到(choose == i)，则更新finish[i]和work[i]；若未找到(choose == -1)，则遍历检查当前是否还有未完成的进程。若有，则说明有进程被死锁，新状态为非安全状态，请求不能被批准；若没有，则说明所有进程都可以完成，新状态为安全状态，请求能被批准，所以用新状态更新当前状态。

2、release\_resource()

首先根据custom\_num和release[]检查是否有足够的资源能被释放：如果release > allocate，则释放指令无法执行。如果不存在以上情况，则更新available[]、allocation[][]、need[][]。

3、show\_current\_state()

打印当前的allocation[][]、maximum[][]、available[]、need[][]。

三、主函数

1、从命令行参数获取available[]

2、从maximum.txt文件读取maximum[][]

3、初始化allocation[][]和need[]

4、仿照project2-1：Unix Shell写命令行“banker>”主循环代码框架。

5、input[]存储输入的指令。

6、RQ指令

处理input[]得到custom\_num和request[]，调用函数request\_resources()，根据返回值输出信息。

7、RL指令

处理input[]得到custom\_num和release[]，调用函数release\_resources()。

8、\* 指令

调用函数show\_current\_state()。

9、exit指令

退出banker命令行，即程序终止

（三）运行结果

