Banker’s Algorithm

根据书本中的要求，该程序将以命令行的形式传递每种类型的资源数量来调用程序。现以规定共4种资源，则可用main函数的argv[]数组承接数据，并用其初始化available数组。

与available数组初始化不同的是，代表每个任务所需要的最大资源数和种类的maximum数组则是由输入文件max\_request.txt来决定的。用FILE类型变量max\_req来承接输入文件，依次读入其中的数据，初始化maximum二维数组。

然后就是通过指令去控制资源的的申请与释放了。

首先用fgets()从控制台读取指令，然后识别。

增添一个exit指令用于结束程序，接收到此指令时，程序即跳出循环，结束进程。

对于\*指令，则调用display\_all()函数，将所有任务、所有资源的Alloction、Max、Need、Available情况按顺序打印到控制台中。仅展示，不做修改，不改变安全性。

对于RL指令，在读取了释放资源的任务编号customer\_num和各个资源的释放量resources[]之后，调用release\_resources()函数。该函数需判断命令释放的资源是否超出可以释放的资源总量，若超出总量，则仅能将所有可释放的资源释放，并报错。因为释放资源不会导致原本出于安全状态的系统变成不安全，故不需要检测经过释放操作后系统的安全性。

对于RQ指令，在读取了申请资源的任务编号customer\_num和各个资源的释放量resources[]之后，调用request\_resources()函数。该函数首先需判断该任务的资源申请是否超出了可申请的资源总量（即available），以及是否超出了其实际需要的资源总量（即need），若是超出，则报错。

若无以上错误，则先尝试允许申请，即改变该任务对应的allocation和need，以及available数组。然后调用security\_detection()函数去检验修改后系统的安全性，

若安全（security\_detection()返回0）则该申请安全，允许通过；

若不安全（security\_detection()返回-1）则该申请不安全，不允许通过，则在报错后，改变该任务对应的allocation和need，以及available数组，将系统还原。

Security\_detection()函数如何检验系统的安全性？即使用安全算法。

1. 令work和finish分别为长度的资源个数m和任务个数n的向量，对于i=0，1，~，n-1，初始化work=available和finish=false
2. 查找这样的i使其满足
3. finish[i]=false
4. need[i]<work

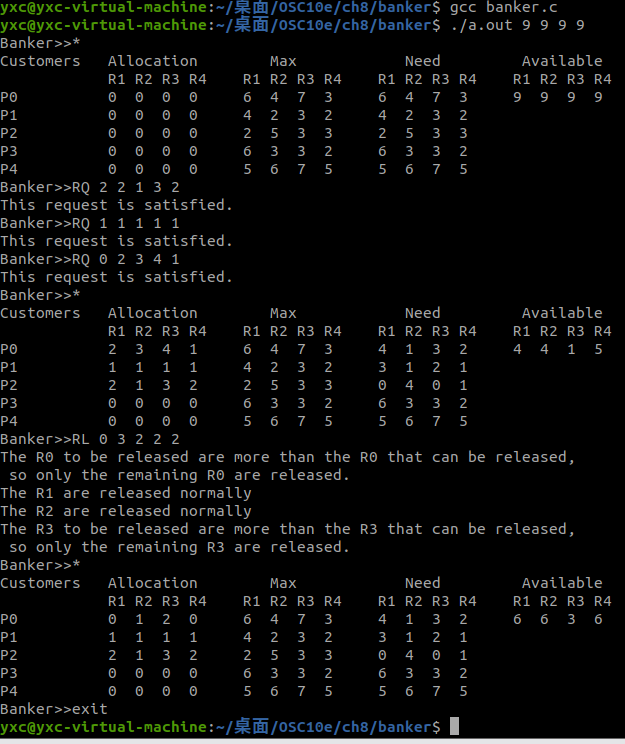
如果没有这样的i存在，那么转到第4步

1. work=work+allocation[i]

finish[i]=true

返回到第2步

1. 如果对所有的i，finish[i]=true，那么系统处于安全状态。

实验结果如图