电子科技大学

计算机专业类课程

实验报告

课程名称：操作系统

学　　院：计算机学院

专　　业：计算机科学与技术

学生姓名：冯铭扬

学　　号：2013060109023

指导教师：薛瑞尼

日　　期：　　年　　月　　日

电 子 科 技 大 学

实　　验　　报　　告

实验二

一、实验名称：银行家算法程序

二、实验学时：4

三、实验内容和目的：

根据要求编写银行家算法程序

四、实验原理：

银行家算法是一个避免死锁（Deadlock）的著名算法，是由艾兹格·迪杰斯特拉在 1965 年为 T.H.E 系统设计的一种避免死锁产生的算法。它以银行借贷系统的分配策略为基础，判断并保证系统的安全运行。

五、实验器材（设备、元器件）

Mac＋xcode

六、实验数据及结果分析：

```
资源管理系统
请输入系统可供资源种类的数量:3
资源1的名称:a
资源的数量:2
资源2的名称:b
资源的数量:3
资源3的名称:c
资源的数量:1

请输入作业的数量:4
请输入各进程的最大需求量(4*3矩阵)[Max]:
1
2 4 4
4 3
 3 3
2
2 3
3
请输入各进程已经申请的资源量(4*3矩阵)[Allocation]:
1 2 1 5 3 3 4 5 5 4 3 4 2
申请的资源大于最大需求量，请重新输入!
请输入各进程已经申请的资源量(4*3矩阵)[Allocation]:
1 3 2 1 2 2 2 1 2 0 1 2 1
申请的资源大于最大需求量，请重新输入!
请输入各进程已经申请的资源量(4*3矩阵)[Allocation]:
1 2 1 1 2 3 1 1 0 0 2 1 1
申请的资源大于最大需求量，请重新输入!
请输入各进程已经申请的资源量(4*3矩阵)[Allocation]:
 1 1 0 1 0 0 0 1 0 0 0 0 0
申请的资源大于最大需求量，请重新输入!
请输入各进程已经申请的资源量(4*3矩阵)[Allocation]:
0 0 0 0 0 0 0 0 0 0 0 0 0
系统目前可用的资源[Avaliable]:
a b c
2 3 1
              Max       Allocation      Need
进程名       a b c        a b c        a b c
 0           1 2 4        0 0 0        1 2 4
 1           4 4 3        0 0 0        4 4 3
 2           3 3 2        0 0 0        3 3 2
 3           2 3 3        0 0 0        2 3 3
产生死锁
是否用银行家算法演示 0否 1是
Program ended with exit code: 1
```

实验代码

```
//
//  main.cpp
//  lab-2
//
//  Created by 冯铭扬 on 16/6/10.
//  Copyright © 2016年 冯铭扬. All rights reserved.
//

#include<iostream>
#include<string.h>
#include<stdio.h>
#define False 0
#define True 1
int Max[100][100]={0};//各进程所需各类资源的最大需求

int Avaliable[100]={0};//系统可用资源

char name[100]={0};//资源的名称

int Allocation[100][100]={0};//系统已分配资源

int Need[100][100]={0};//还需要资源

int Request[100]={0};//请求资源向量

int temp[100]={0};//存放安全序列

int Work[100]={0};//存放系统可提供资源

int M=100;//作业的最大数为100

int N=100;//资源的最大数为100


int changedata(int i)//进行资源分配

{
    int j;
    for (j=0;j<M;j++) {
        Avaliable[j]=Avaliable[j]-Request[j];
        Allocation[i][i]=Allocation[i][i]+Request[i];
```

```
        Need[i][j]=Need[i][j]-Request[j];
    }
    return 1;
}

int safe()//安全性算法

{
    int i,k=0,m,apply,Finish[100]={0};
    int j;
    int flag=0;
    Work[0]=Avaliable[0];
    Work[1]=Avaliable[1];
    Work[2]=Avaliable[2];
    for(i=0;i<M;i++){
        apply=0;
        for(j=0;j<N;j++){
            if (Finish[i]==False&&Need[i][j]<=Work[j]){
                apply++;
                if(apply==N){
                    for(m=0;m<N;m++)

                        Work[m]=Work[m]+Allocation[i][m];//变分配数

                    Finish[i]=True;
                    temp[k]=i;
                    i=-1;
                    k++;
                    flag++;
                }
            }
        }
    }
    for(i=0;i<M;i++){
        if(Finish[i]==False){

            std::cout<<"产生死锁"<<std::endl;//产生了死锁

            return -1;
        }
    }

    std::cout<<"系统是安全的!"<<std::endl;//如果安全，输出成功

    std::cout<<"分配的序列:";

    for(i=0;i<M;i++){//输出运行进程数组
```

```
        std::cout<<temp[i];
        if(i<M-1) std::cout<<"->";
    }
    std::cout<<std::endl;
    return 0;
}

void showMatrix()//显示资源矩阵

{

    int i,j;

    std::cout<<"系统目前可用的资源[Avaliable]:"<<std::endl;

    for(i=0;i<N;i++)
        std::cout<<name[i]<<" ";
    std::cout<<std::endl;
    for (j=0;j<N;j++)

        std::cout<<Avaliable[j]<<" ";//输出分配资源

    std::cout<<std::endl;
    std::cout<<"           Max     Allocation    Need"<<std::endl;

    std::cout<<"进程名     ";

    for(j=0;j<3;j++)

    {
        for(i=0;i<N;i++)
            std::cout<<name[i]<<" ";
        std::cout<<"     ";

    }
    std::cout<<std::endl;
    for(i=0;i<M;i++){
        std::cout<<" "<<i<<"        ";
        for(j=0;j<N;j++)
            std::cout<<Max[i][j]<<" ";
        std::cout<<"     ";
        for(j=0;j<N;j++)
            std::cout<<Allocation[i][j]<<" ";
        std::cout<<"     ";
        for(j=0;j<N;j++)
            std::cout<<Need[i][j]<<" ";
        std::cout<<std::endl;
    }
}

void share()//利用银行家算法对申请资源对进行判定
```

```
{
    char ch;
    int i=0,j=0;
    ch='y';
    std::cout<<"请输入要求分配的资源进程号(0-"<<M-1<<"):";

    std::cin>>i;//输入须申请的资源号

    std::cout<<"请输入进程 "<<i<<" 申请的资源:"<<std::endl;
    for(j=0;j<N;j++)
    {
        std::cout<<name[j]<<":";
        std::cin>>Request[j];//输入需要申请的资源
    }
    for (j=0;j<N;j++){
        if(Request[j]>Need[i][j])//判断申请是否大于需求，若大于则出错
        {
            std::cout<<"进程 "<<i<<"申请的资源大于它需要的资源";

            std::cout<<" 分配不合理，不予分配！"<<std::endl;
            ch='n';
            break;
        }
        else {
            if(Request[j]>Avaliable[j])//判断申请是否大于当前资源，若大于则出错
            {
                std::cout<<"进程"<<i<<"申请的资源大于系统现在可利用的资源";

                std::cout<<" 分配出错，不予分配!"<<std::endl;
                ch='n';
                break;
            }
        }
    }
    if(ch=='y') {
        changedata(i);//根据进程需求量变换资源
```

```
        showMatrix();//根据进程需求量显示变换后的资源

        safe();//根据进程需求量进行银行家算法判断

    }
}

void addresources(){//添加资源

    int n,flag;

    std::cout<<"请输入需要添加资源种类的数量:";

    std::cin>>n;
    flag=N;
    N=N+n;
    for(int i=0;i<n;i++){

        std::cout<<"名称:";

        std::cin>>name[flag];

        std::cout<<"数量:";

        std::cin>>Avaliable[flag++];
    }
    showMatrix();
    safe();
}

void delresources(){//删除资源

    char ming;
    int i,flag=1;

    std::cout<<"请输入需要删除的资源名称：";

    do{
        std::cin>>ming;
        for(i=0;i<N;i++)
            if(ming==name[i]){
                flag=0;
                break;
            }
        if(i==N)

            std::cout<<"该资源名称不存在，请重新输入：";

    }
    while(flag);
    for(int j=i;j<N-1;j++)
    {
        name[j]=name[j+1];
        Avaliable[i]=Avaliable[i+1];
```

```
    }
    N=N-1;
    showMatrix();
    safe();
}

void changeresources(){//修改资源函数

    std::cout<<"系统目前可用的资源:"<<std::endl;

    for(int i=0;i<N;i++)
        std::cout<<name[i]<<":"<<Avaliable[i]<<std::endl;

    std::cout<<"输入系统可用资源:"<<std::endl;

    std::cin>>Avaliable[0]>>Avaliable[1]>>Avaliable[2];

    std::cout<<"经修改后的系统可用资源为"<<std::endl;

    for (int k=0;k<N;k++)
        std::cout<<name[k]<<":"<<Avaliable[k]<<std::endl;
    showMatrix();
    safe();
}

void addprocess(){//添加作业

    int flag=M;
    M=M+1;

    std::cout<<"请输入该作业的最大需求量"<<std::endl;

    for(int i=0;i<N;i++){
        std::cout<<name[i]<<":";
        std::cin>>Max[flag][i];
        Need[flag][i]=Max[flag][i]-Allocation[flag][i];
    }
    showMatrix();
    safe();
}

int main()//主函数

{


    int i,j,number,choice,m,n,flag;
    char ming;

    std::cout<<"资源管理系统"<<std::endl;
```

```
    std::cout<<"请输入系统可供资源种类的数量:";
    std::cin>>n;
    N=n;
    for(i=0;i<n;i++)
    {
        std::cout<<"资源"<<i+1<<"的名称:";
        std::cin>>ming;
        name[i]=ming;
        std::cout<<"资源的数量:";
        std::cin>>number;
        Avaliable[i]=number;
    }
    std::cout<<std::endl;
    std::cout<<"请输入作业的数量:";
    std::cin>>m;
    M=m;
    std::cout<<"请输入各进程的最大需求量("<<m<<"*"<<n<<"矩
阵)[Max]:"<<std::endl;
    for(i=0;i<m;i++)
        for(j=0;j<n;j++)
            std::cin>>Max[i][j];
    do{
        flag=0;
        std::cout<<"请输入各进程已经申请的资源量("<<m<<"*"<<n<<"矩
阵)[Allocation]:"<<std::endl;
        for(i=0;i<m;i++)
            for(j=0;j<n;j++){
                std::cin>>Allocation[i][j];
                if(Allocation[i][j]>Max[i][j])
                    flag=1;
                Need[i][j]=Max[i][j]-Allocation[i][j];
            }
        if(flag)
            std::cout<<"申请的资源大于最大需求量，请重新输入!\n";
    }
    while(flag);
```

```
showMatrix();//显示各种资源

safe();//用银行家算法判定系统是否安全

std::cout<<"是否用银行家算法演示 0否 1是"<<std::endl;

std::cin>>choice;
while(choice)
{
    std::cout<<"            银行家算法演示   "<<std::endl;

    std::cout<<"              1:增加资源    "<<std::endl;

    std::cout<<"              2:删除资源    "<<std::endl;

    std::cout<<"              3:修改资源    "<<std::endl;

    std::cout<<"              4:分配资源    "<<std::endl;

    std::cout<<"              5:增加作业    "<<std::endl;

    std::cout<<"              0:离开        "<<std::endl;

    std::cout<<"请选择功能号: ";

    std::cin>>choice;
    switch(choice)
    {
        case 1: addresources();break;
        case 2: delresources();break;
        case 3: changeresources();break;
        case 4: share();break;
        case 5: addprocess();break;
        case 0: choice=0;break;

        default: std::cout<<"请正确选择功能号(0-5)!"<<std::endl;break;

    }
}
return 1;

}
```

电 子 科 技 大 学

实　　验　　报　　告

实验二

一、实验名称：

二、实验学时：4

三、实验内容和目的：

四、实验原理：

(按实验内容分析实验原理并填写)

五、实验器材（设备、元器件）

六、实验步骤：

七、实验数据及结果分析：

(按实验步骤顺序填写代码、数据或截图)

八、实验结论、心得体会和改进建议：

九、