# 银行家算法《操作系统》课程设计报告

来源：网络 作者：岁月静好 更新时间：2024-10-17

*《操作系统》课程设计报告课题：银行家算法专业计算机科学与技术学生姓名班级计算机学号指导教师信息工程学院一、实验要求和实验目的实验目的：本课程设计是学生学习完《操作系统原理》课程后，进行的一次全面的综合训练，通过课程设计，让学生更好地掌握操作...*

《操作系统》课程设计报告

课题：

银行家算法

专业

计算机科学与技术

学生姓名

班级

计算机

学号

指导教师

信息工程学院

一、实验要求和实验目的实验目的：本课程设计是学生学习完《操作系统原理》课程后，进行的一次全面的综合训练，通过课程设计，让学生更好地掌握操作系统的原理及实现方法，加深对操作系统基础理论和重要算法的理解，加强学生的动手能力。

实验要求：从课程设计的目的出发，通过设计工作的各个环节，达到以下教学要求：两人一组，每组从所给题目中任选一个（如自拟题目，需经指导教师同意），每个学生必须独立完成课程设计，不能相互抄袭，同组者文档不能相同；设计完成后，将所完成的工作交由指导教师检查；要求写出一份详细的设计报告。

二、设计内容：

课题一、编制银行家算法通用程序，并检测所给状态的系统安全性。

1）银行家算法中的数据结构：

可利用资源向量Available。这是一个含有m个

元素的数组，其中的每一个元素代表一类可利用的资源数目，其初始值是系统中所配置的该类全部可用资源的数目，其数值随该类资源的分配和回收而动态地改变。Available[j]=K，则表示系统中现有Rj

类资源K个。

最大需求矩阵Max。这是一个n\*m的矩阵，它定义了系统中n个进程中的每一个进程对m类资源的最大需求。如果Max[i，j]=K，则表示进程i需要Rj类资源的最大数目为K。

1.分配矩阵Allocation。这也是一个n\*m的矩阵，它定义了系统中每一类资料当前已分配给没一进程的资源数。如果Allocation[i，j]=K，则表示进程i当前已分得Rj类资源的数目为K。需求矩阵Need。这也是一个n\*m的矩阵，用以表示每一个进程尚需的各类资源数。如果Need[i,j]=K，则表示进程i还需要Rj类资源K个，方能完成其任务。

上述三个矩阵存在如下关系：

Need[i,j]=

Max[i，j]-

Allocation[i，j]

2）银行家算法

设Request[i]

是进程Pi的请求向量，如果Request[i，j]=K,表示进程Pi需要K个Rj类型的资源。当Pi发出资源请求后，系统按下述步骤进行检查：如果Request[i，j]

#include

#include

#include

#include

#include

const

int

MAX\_P=20;

const

int

MAXA=10;

//定义A类资源的数量

const

int

MAXB=5;

const

int

MAXC=7;

typedef

struct

node{

int

a;

int

b;

int

c;

int

remain\_a;

int

remain\_b;

int

remain\_c;

}bank;

typedef

struct

node1{

char

name[20];

int

a;

int

b;

int

c;

int

need\_a;

int

need\_b;

int

need\_c;

}process;

bank

banker;

process

processes[MAX\_P];

int

quantity;

//初始化函数

void

initial()

{

int

i;

banker.a=MAXA;

banker.b=MAXB;

banker.c=MAXC;

banker.remain\_a=MAXA;

banker.remain\_b=MAXB;

banker.remain\_c=MAXC;

for(i=0;i>name;

for(i=0;i>need\_a;

cout>need\_b;

cout>need\_c;

t=1;

coutbanker.remain\_a){

coutbanker.remain\_b){

coutbanker.remain\_c){

cout>name;

p=-1;

for(i=0;i>a;

cout>b;

cout>c;

flag=1;

if((a>banker.remain\_a)||(a>processes[p].need\_a-processes[p].a)){

coutbanker.remain\_b)||(b>processes[p].need\_b-processes[p].b)){

coutbanker.remain\_c)||(c>processes[p].need\_c-processes[p].c)){

cout>name;

p=-1;

for(i=0;i0){

for(i=0;i>chioce;

switch(chioce){

case

1:

add();

break;

case

2:

bid();

break;

case

3:

finished();

break;

case

4:

view();

break;

case

0:

flag=0;

break;

default:

cout<<“选择错误“<<endl<<endl;

}

}

}

六、使用说明

运行环境C-FREE4.0，新建任务。将编制好的代码输入此运行环境中。

按F5：出现如上图所示窗口。按照提示，新建一个作业：wujun。为作业分配资源，A：3；B：4；C：5。输入2，为作业分配资源。三种资源的数量分配分别为:A:3;B:5;C:4。输入4，查看资源情况。出现出错提示，所申请的B类资源超过银行家所剩B类资源或作业申请资源失败。输入0，退出系统。

重新加入一个作业：wujun1.并为作业分配资源分别为A:3;B:3;C:3，为该作业分配资源A:3;B:2;C:2.输入4查看资源情况。

显示输出，银行家算法所剩资源（剩余资源、总共资源）。

七、实验心得

八、参考文献

汤子瀛等.计算机操作系统.西安电子科技大学出版社.2024年5月

蒋静

徐志伟.操作系统原理•技术与编程『M』.北京:机械工业出版社,2024

本文档由站牛网zhann.net收集整理，更多优质范文文档请移步zhann.net站内查找