课程名称 计算机操作系统

实验成绩

指导教师 曹勇



实 验 报 告

院系 信息工程学院

班级 物联网工程

学号 1601141024

姓名 李程里

日期 2018/10/31 地点：505

**­** 实验五、FIFIO算法模拟

### 实验五、FIFO算法模拟

一、实验目的

一个作业有多少个进程，处理机只分配固定的主存页面供该作业执行。往往页面数小于进程数，当请求调页程序调进一个页面时，可能碰到主存中并没有空闲块的情况，此时就产生了在主存中淘汰哪个页面的情况。本实验要求模拟FIFO算法

二、实验原理

此算法的实质是，总是选择在主存中居留最长时间的页面淘汰。理由是：最早调入主存的页，其不再被访问的可能性最大。

三、实验环境

1、pc

2、vc++

四、程序源代码：

#define MAXSIZE 20

#include <iostream>

using namespace std;

int main()

{

int label=0; //标记此页是否已经装入内存

int input=0; //用于输入作业号

int worknum=0; //记录作业个数

int storesize=0; //系统分配的存储块数

int interrupt=0; //中断次数

int quence[MAXSIZE]; //队列FIFO算法的主要数据结构

int workstep[MAXSIZE]; //用于记录作业走向

/\*

初始化

\*/

for(int i=0;i<MAXSIZE;i++) {

quence[i]=0;

workstep[i]=0;

}

cout<<"请输入存储区块数：";

cin>>storesize;

cout<<"请输入作业走向（输入0结束）：\n";

for(int j=0;j<MAXSIZE;j++) {

cout<<"页面号："<<j+1<<" ";

cin>>input;

workstep[j]=input;

if(input==0)

{

cout<<"输入结束！\n";

break;

}

worknum++;

}

if(workstep[0]==0)

{

cout<<"未输入任何作业，系统将退出！\n";

return 0;

}

cout<<"置换情况如下：\n";

for(int k=0;k<worknum;k++)

{

label=0;

/\*

看队列中是否有相等的页号或空位置

\*/

for(int l=0;l<storesize;l++) {

/\*

是否有相等的页号

\*/

if(quence[l]==workstep[k]){

cout<<"内存中有"<<workstep[k]<<"号页面，无须中断！ \n";

label=1; //标记此页面已装入内存

break;

}

/\*

是否有空位置

\*/

if(quence[l]==0){

quence[l]=workstep[k];

cout<<"发生中断,但内存中有空闲区，"<<workstep[k]<<"号页面直接调入！\n";

interrupt++;

label=1;

break;

}

}

/\*

上述情况都不成立则调出对首，将调入页面插入对尾

\*/

if(label==0) {

cout<<"发生中断，将"<<quence[0]<<"号页面调出，"<<workstep[k]<<"号装入！\n";

interrupt++;

for(int m=0;m<storesize;m++)

{

quence[m]=quence[m+1];

}

quence[storesize-1]=workstep[k];

}

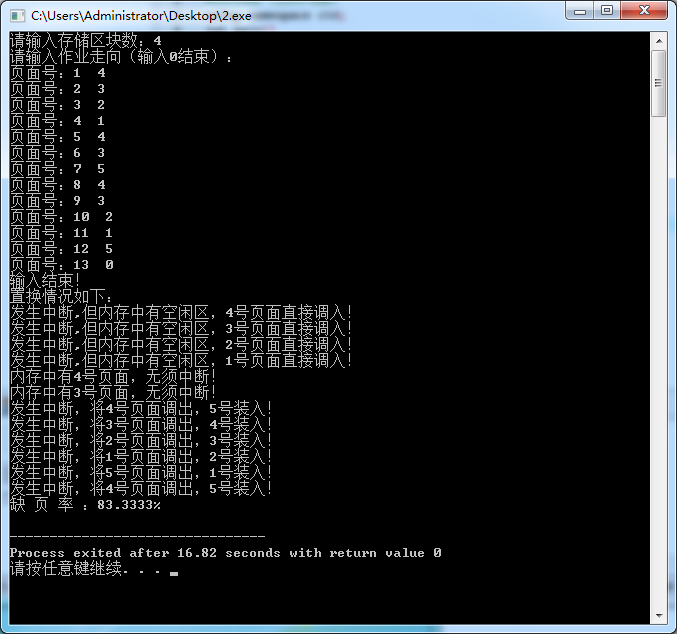
}

cout<<"缺 页 率 ："<<float(interrupt)/float(worknum)\*100<<"%\n";

}

五、实验程序截图





六、实验总结

本次实验，让我深刻理解了FIFO算法，与上次LRU算法实验相比较。由于FIFO所依据的条件是各个页面存入的时间，而也页面调入先后并不能反映页面的使用情况，所以LRU算法相对较好。也通过本次实验让我体会了编程思想的作用，一个程序的成功取决于思想的设计。