**实验一 进程调度模拟算法实验报告**

课程名称 操作系统 信息技术与科学 分院信1605-3 班 组 桌号

实验者姓名 杨蕾 实验日期 2019 年 4 月 17 日

评分 教师签名

一、实验目的

进程调度是处理机管理的核心内容。本实验要求用高级语言编写模拟进程调度程序，以便加深理解有关进程控制快、进程队列等概念，并体会和了解优先数算法和时间片轮转算法的具体实施办法。

二、实验要求

1、设计进程控制块 PCB 的结构，通常应包括如下信息：

进程名、进程优先数（或轮转时间片数）、进程已占用的 CPU 时间、进程到完成还需要的时间、进程的状态、当前队列指针等。

2、编写两种调度算法程序：优先数调度算法程序、循环轮转调度算法程序。

3.按要求输出结果。

三、实验过程

1． 准备

A． 查阅相关资料；

通过看书对优先数调度算法及循环轮转调度算法有更深的理解。

B． 初步编写程序；

精读提示与说明，创建调度算法，对PCB进行设计结构，对7个过程分别进行设计与实现。

C． 准备测试数据；

分别对优先数调度算法及循环轮转调度算法准备测试数据。

如(a,5),(b,3),(c,1),(d,6)

D． 运行。

2． 上机调试。

对优先及轮转法分别进行调试。

3． 主要流程和源代码。

**主要流程：**

建立PCB结构及其他有关变量

创建输出PCB函数print(PCB \*run, PCB \*ready, PCB \*finish)、insert1(PCB \*p,PCB \*ready)将未完成的PCB按优先顺序插入到就绪队列，insert2(PCB \*p,PCB \*ready)轮转法插入，insert3(PCB \*p,PCB \*ready)插入完成队列，create1(int n,PCB \*ready)创建优先队列，create2(int n,PCB \*ready)轮转法创建插入就绪队列，firstin1(int time, PCB \*ready, PCB \*run, PCB \*finish) 优先调度调度就绪队列的第一个进程投入运行，firstin2(int time, PCB \*ready, PCB \*run, PCB \*finish)时间片轮转法调度就绪队列的第一个进程投入运行，prisch(PCB \*ready, PCB \*run, PCB \*finish) 按优先数算法调度进程，roundsch(PCB \*ready, PCB \* run, PCB \*finish) 按时间片轮转法调度进程。

使用main函数将算法连接起来。

**源代码：**

#include<iostream>

using namespace std;

struct PCB {

char name[20];

int prio;

int round;

int cputime;

int needtime;

bool flag;

char state;

PCB\* next;

};

//显示每执行一次后所有进程的状态及有关信息

void print(PCB \*run, PCB \*ready, PCB \*finish)

{

cout << endl << "name" << "\t" << "cputime" << "\t" << "needtime" << "\t" << "priority" << "\t" << "state"<< endl;

cout << endl << "正在运行:" << endl;

if (run)

{

cout << run->name << "\t" << run->cputime << "\t" << run->needtime << "\t\t" << run->prio << "\t\t" << run->state << endl;

run = NULL;

}

cout << endl << "就绪队列:" << endl;

while (ready->next)

{

if (ready->next)

{

cout << ready->next->name << "\t" << ready->next->cputime << "\t" << ready->next->needtime << "\t\t" << ready->next->prio << "\t\t" << ready->next->state << endl;

ready = ready->next;

}

}

cout << endl << "完成队列:" << endl;

while (finish->next)

{

if (finish->next)

{

cout << finish->next->name << "\t" << finish->next->cputime << "\t" << finish->next->needtime << "\t\t" << finish->next->prio << "\t\t" << finish->next->state << endl;

finish = finish->next;

}

}

cout << endl;

system("pause");

}

//在优先数算法中，将尚未完成的PCB 按优先数顺序插入到就绪队列中

void insert1(PCB \*p, PCB \*ready)

{

PCB \*q, \*temp;

q = new PCB;

q->next = NULL;

temp = new PCB;

temp->next = NULL;

temp = ready;

q = ready->next;

while (q && (q->prio) >= (p->prio))

{

temp = q;

q = q->next;

}

if (!q)

{

p->state = 'W';

temp->next = p;

p->next = NULL;

}

else

{

p->state = 'W';

p->next = q;

temp->next = p;

}

}

//轮转法插入

void insert2(PCB \*p, PCB \*ready)

{

PCB \*q;

q = new PCB;

q->next = NULL;

q = ready;

while (q->next)

{

q = q->next;

}

p->state = 'W';

q->next = p;

p->next = NULL;

}

//插入完成队列

void insert3(PCB \*p, PCB \*finish)

{

PCB \*q;

q = new PCB;

q->next = NULL;

q = finish;

while (q->next)

{

q = q->next;

}

q->next = p;

p->state = 'F';

p->next = NULL;

}

//创建新进程，并将它的PCB插入就绪队列

void create1(int n, PCB \*ready)

{

PCB \*p;

for (int i = 0; i < n; i++)

{

p = new PCB;

p->next = NULL;

cout << endl << "进程名称：";

cin >> p->name;

cout << endl << "该进程执行时间：";

cin >> p->needtime;

p->cputime = 0;

p->prio = 50 - p->needtime;

p->round = 0;

p->flag = false;

p->state = 'W';

insert1(p, ready);

}

}

//轮转法，并将它的PCB插入就绪队列

void create2(int n, PCB \*ready)

{

PCB \*p;

for (int i = 0; i < n; i++)

{

p = new PCB;

p->next = NULL;

cout << endl << "进程名称：";

cin >> p->name;

cout << endl << "该进程执行时间：";

cin >> p->needtime;

p->cputime = 0;

p->prio = 0;

p->round = 2;

p->flag = false;

p->state = 'W';

insert2(p, ready);

}

}

//优先调度调度就绪队列的第一个进程投入运行

void firstin1(int time, PCB \*ready, PCB \*run, PCB \*finish)

{

run = NULL;

while (ready->next)

{

if (ready->next)

{

run = ready->next;

ready->next = run->next;

run->next = NULL;//移出就绪队列

run->state = 'R';

run->prio--;

if (!run->flag)

{

run->flag = true;

}

run->cputime++;

run->needtime--;

time++;//运行

if (run->needtime <= 0)

{

run->needtime = 0;

}

print(run, ready, finish);

if (run->needtime <= 0)

{

insert3(run, finish);//插入完成队列

run = NULL;

print(run, ready, finish);

firstin1(time, ready, run, finish);

}

else

{

insert1(run, ready);//插入就绪队列

run = NULL;

print(run, ready, finish);

firstin1(time, ready, run, finish);

}

}

else

{

run = NULL;

}

}

}

//时间片轮转法调度就绪队列的第一个进程投入运行

void firstin2(int time, PCB \*ready, PCB \*run, PCB \*finish)

{

run = NULL;

while (ready->next)

{

if (ready->next)

{

run = ready->next;

ready->next = run->next;

run->next = NULL;//移出就绪队列

run->state = 'R';

if (!run->flag)

{

run->flag = true;

}

if (run->needtime >= run->round)

{

run->cputime += run->round;

}

else

{

run->cputime += run->needtime;

}

if (run->needtime >= run->round)

{

time += run->round;

run->needtime -= run->round;

}

else

{

time += run->needtime;

run->needtime = 0;

}//运行

print(run, ready, finish);

if (run->needtime <= 0)

{

insert3(run, finish);//插入完成队列

run = NULL;

print(run, ready, finish);

firstin2(time, ready, run, finish);

}

else

{

insert2(run, ready);//插入就绪队列

run = NULL;

print(run, ready, finish);

firstin2(time, ready, run, finish);

}

}

else

{

run = NULL;

}

}

}

//按优先数算法调度进程

void prisch(PCB \*ready, PCB \*run, PCB \*finish)

{

int n, time = 0;

cout << endl << "请输入进程个数：";

cin >> n;

create1(n, ready);

print(run, ready, finish);

firstin1(time, ready, run, finish);

}

//按时间片轮转法调度进程

void roundsch(PCB \*ready, PCB \* run, PCB \*finish)

{

int n, time = 0;

cout << endl << "请输入进程个数：";

cin >> n;

create2(n, ready);

print(run, ready, finish);

firstin2(time, ready, run, finish);

}

void main()

{

while (true)

{

PCB \*run, \*ready, \*finish;

run = new PCB;

run = NULL;

ready = new PCB;

ready->next = NULL;

finish = new PCB;

finish->next = NULL;

int choice;

cout << endl << "1.优先级调度算法\t2.轮转调度算法" << endl<< "请选择算法:";

cin >> choice;

switch (choice)

{

case 1:

prisch(ready, run, finish);

break;

case 2:

roundsch(ready, run, finish);

break;

default:

cout << endl << "输入有误, 请重新输入" << endl;

break;

}

delete finish, ready, run;

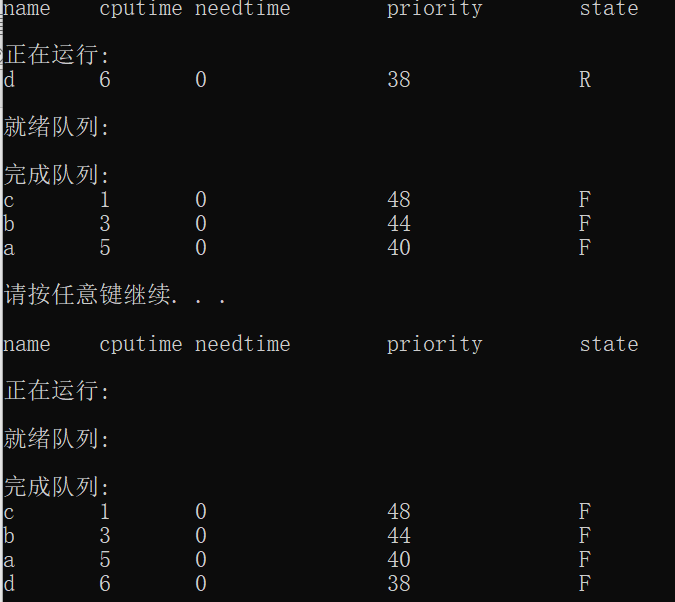
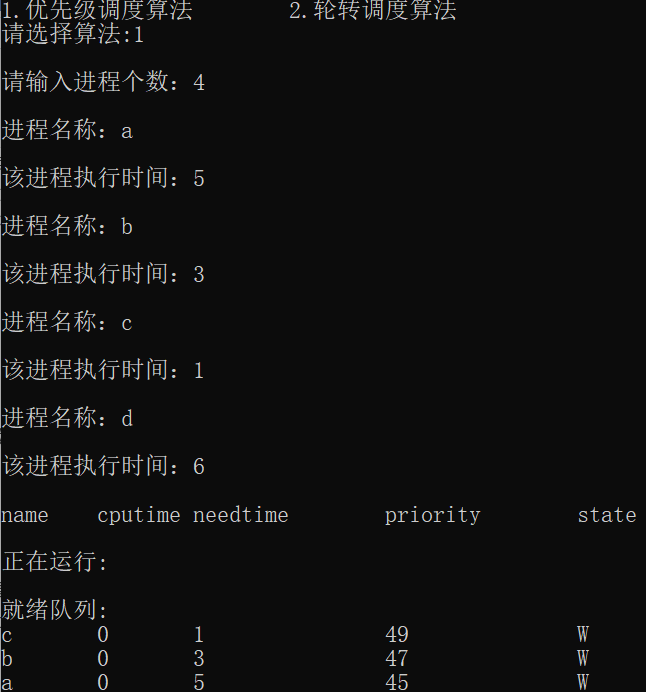
}}

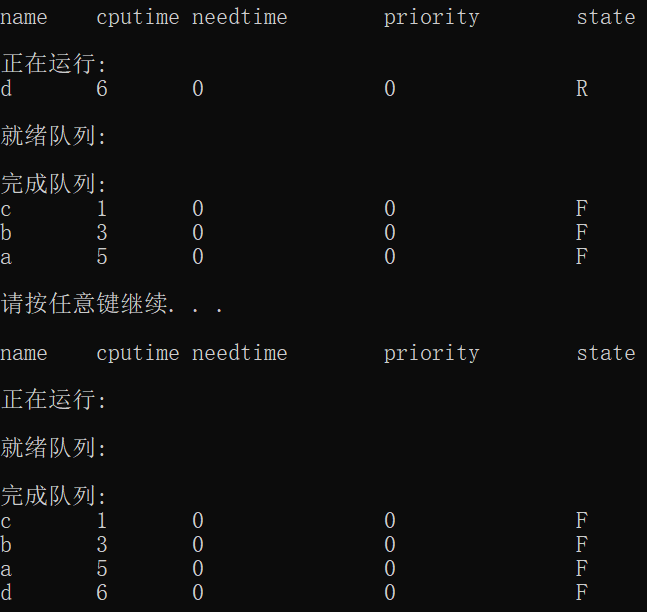
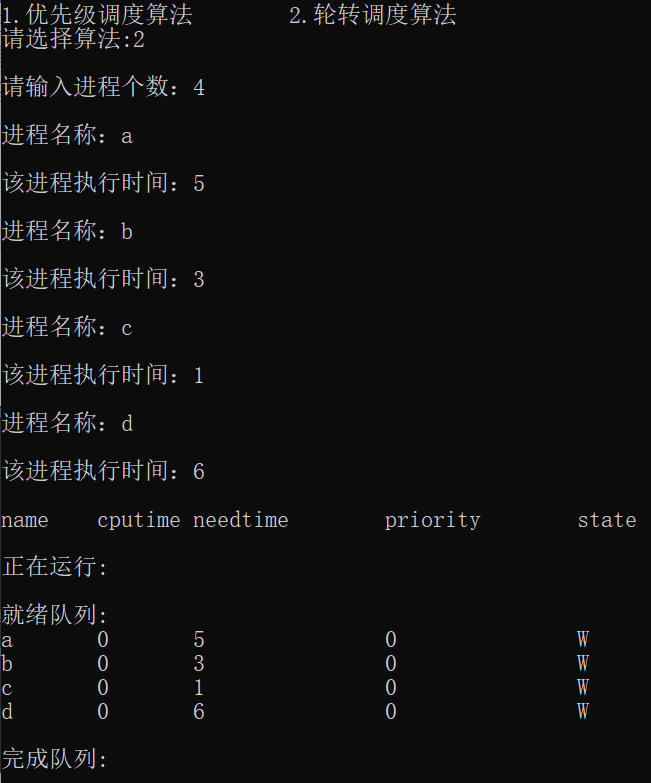
4． 遇到的主要问题和解决方法

A．结构体中string输入报错，解决方法为改为char数组输入；明白了结构名中只有五种数据类型(整型、浮点型、字符型、指针型和 无值型)。

B．对链表的创建插入困惑，解决方法为通过复习数据结构，重新熟悉链表的创建插入。

四、实验结果





五、实验总结

通过本次实验，我学到了优先数调度算法及循环轮转调度算法在代码中的实现方法，了解了struct的使用，掌握了将优先数最高的数据放到队列前面，提高了对数据结构的理解力与编程能力，对队列的创建、插入更加熟悉。