课程名称 操作系统

实验者姓名 孟祥通 实验日期 2017 年 5 月 4 日

1. **实验目的**

通过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。

1. **实验要求**
2. 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。

1. 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

1. 按要求输出结果。
2. **实验过程**
3. 算法设计：

在优先数算法中，进程优先数的初值设为：50-NEEDTIME每执行一次，优先数减 1，CPU 时间片数加 1，进程还需要的时间片数减 1。在轮转法中，采用固定时间片单位（两个时间片为一个单位），进程每轮转一次， CPU时间片数加 2，进程还需要的时间片数减 2，并退出 CPU，排到就绪队列尾，等待下一次调度。

1. 程序实现：

优先数调度算法程序：

#include<iostream>

#include<string>

using namespace std;

typedef struct pcb{

string name;

int priority; //进程优先级

int needtime; //进程到完成还需要的时间

int cputime; //进程已占用cpu时间

string state;

struct pcb\* next;

}\*pPCB;

typedef struct{

pPCB run;

pPCB ready;

pPCB tail;

pPCB finish;

}pcbList;

void PRINT(pcbList &pList)

{

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

pcb \*p = new pcb;

p = pList.ready->next;

while (p != NULL)

{

cout << p->name << " " << p->priority << " " << p->needtime << " " << p->cputime << " " << p->state << endl;

p = p->next;

}

}

//按进程优先级排序，最高的排在最前面

void INSERT1(pcbList &pList)

{

int prio=0;

pcb \*p = new pcb;

pcb \*p1 = new pcb;

p = pList.ready->next;

while (p!=NULL)

{

if (p->priority > prio&&p->needtime != 0)

{

prio = p->priority;

p1 = p;

}

p = p->next;

}

while (p1->needtime > 0)

{

p1->state = "R";

p1->needtime -= 1;

p1->priority -= 1;

p1->cputime += 1;

if (p1->needtime == 0)

{

p1->state = "F";

}

PRINT(pList);

}

}

void CREATE(pcb \*pcbs, pcbList &pList)

{

pcbs->next = NULL;

pList.tail->next = pcbs;

pList.tail = pcbs;

}

void main()

{

int n=0, i=0; //n个进程

pcbList pList;

pList.ready = pList.tail = new pcb;

pList.tail->next = NULL;

cout << "输入进程数" << endl;

cin >> n;

string name="";

int needtime;

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

{

cout << "输入进程名和相应的 NEEDTIME 值" << endl;

pPCB pcbs = new pcb;

cin >> name;

cin >> needtime;

(\*pcbs).name = name;

(\*pcbs).needtime = needtime;

(\*pcbs).cputime = 0;

(\*pcbs).priority = 50 - needtime;

(\*pcbs).state = "W";

CREATE(pcbs, pList);

}

PRINT(pList);

cout << "/\*\*\*\*\*\*\*\*\*\*\*进程开始\*\*\*\*\*\*\*\*\*\*\*/" << endl;

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

{

INSERT1(pList);

}

}

循环轮转调度算法程序：

#include<iostream>

#include<string>

using namespace std;

typedef struct pcb{

string name;

int needtime; //进程到完成还需要的时间

int cputime; //进程已占用cpu时间

string state;

struct pcb\* next;

}\*pPCB;

typedef struct{

pPCB run;

pPCB ready;

pPCB tail;

pPCB finish;

}pcbList;

void PRINT(pcbList &pList)

{

cout << "name" << " needtime" << " cputime" << " state" << endl;

pcb \*p = new pcb;

p = pList.ready->next;

while (p != NULL)

{

cout << p->name << " " << p->needtime << " " << p->cputime << " " << p->state << endl;

p = p->next;

}

}

//按进程优先级排序，最高的排在最前面

void INSERT1(pcbList &pList)

{

int prio = 0;

pcb \*p = new pcb;

while (pList.ready != pList.tail)

{

p = pList.ready->next;

if (p->needtime > 0)

{

p->state = "R";

p->needtime -= 2;

p->cputime += 2;

if (p->needtime <= 0)

{

p->needtime = 0;

p->cputime -= 1;

p->state = "F";

}

PRINT(pList);

}

if (p->needtime <= 0)

{

pList.ready->next = p->next;

if (pList.tail == p)

pList.tail = pList.ready;

p->next = NULL;

}

else

{

if (pList.tail != p)

{

p->state = "W";

pList.ready->next = p->next;

p->next = NULL;

pList.tail->next = p;

pList.tail = p;

}

}

}

}

void CREATE(pcb \*pcbs, pcbList &pList)

{

pcbs->next = NULL;

pList.tail->next = pcbs;

pList.tail = pcbs;

}

void main()

{

int n = 0, i = 0; //n个进程

pcbList pList;

pList.ready = pList.tail = new pcb;

pList.tail->next = NULL;

cout << "输入进程数" << endl;

cin >> n;

string name = "";

int needtime;

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

{

cout << "输入进程名和相应的 NEEDTIME 值" << endl;

pPCB pcbs = new pcb;

cin >> name >> needtime;

(\*pcbs).name = name;

(\*pcbs).needtime = needtime;

(\*pcbs).cputime = 0;

(\*pcbs).state = "W";

CREATE(pcbs, pList);

}

PRINT(pList);

cout << "/\*\*\*\*\*\*\*\*\*\*\*进程开始\*\*\*\*\*\*\*\*\*\*\*/" << endl;

INSERT1(pList);

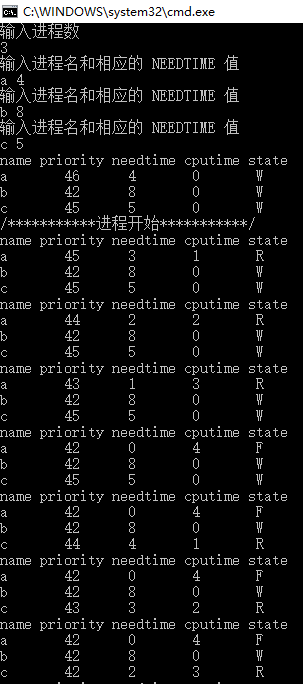
}

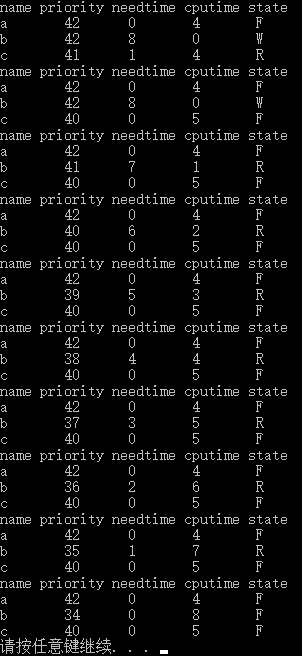
1. 遇到的问题：

在使用队列的时候，对于指针的使用有一些模糊，循环队列的使用有点生疏。

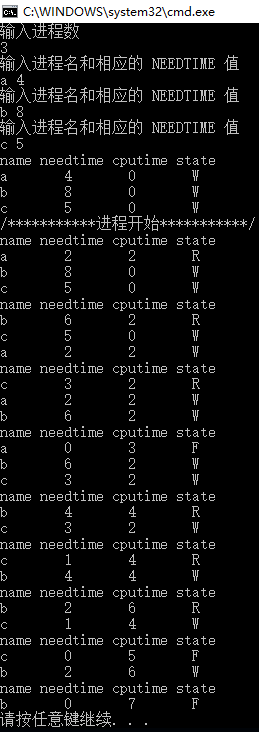
1. **实验结果**

优先数调度算法程序

****

****

循环轮转调度算法程序

****

1. **实验总结**

通过本次实验，我学到了如何利用进程队列来解决优先数算法和时间片轮转算法的具体实施办法。