### C++实验代码

1.main函数：

#include"node.h"

void main()

{

char chose;

number();

getchar();

cout<<"输入进程调度算法（选择P或者R):";

cin>>chose;

switch(chose)

{

case'p':

case'P':

PrioCreate();

Priority();

break;

case'R':

case'r':

TimeCreate();

RoundRun();

break;

default:break;

}

Output();

}

2.node的头文件（node.h）

#include<iostream>

using namespace std;

#include<string>

typedef struct node{

char name[20];//进程的名字

int prio;//进程的优先级

int round;//分配CPU的时间片

int cputime;//CPU执行时间

int needtime;//进程执行所需要的时间

char state;//进程的状态（W-就绪态，R-执行态，F-完成态）

int count;//记录执行的次数

struct node \*next;

}PCB;

void number();

void GetFirst();//从就绪队列获取第一个节点

void Output();//输出队列信息

void InsertPrio(PCB \*in);//创建优先级队列，规定有限数越小优先级越高

void InsertTime(PCB \*in);//时间片队列

void InsertFinish(PCB \*in);//时间片队列

void PrioCreate();//优先级输入函数

void TimeCreate();//时间片输入函数

void Priority();//按照优先级调度

void RoundRun();//时间片轮回转调度

3.node的源文件（node.cpp）

#include"node.h"

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

//定义三个队列，就绪、执行、完成队列

int num;

void GetFirst()

{

run=ready;

if(ready!=NULL)

{

run->state='R';

ready=ready->next;

run->next=NULL;

}

}

void number()

{

cout<<"请输入要创建的进程数目：";

cin>>num;

}

void Output()

{

PCB \*p;

p=ready;

cout<<"进程名 优先级 轮数 时间 需要时间 进程状态 计数器"<<endl;

while(p!=NULL)

{

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

p=p->next;

}

p=finish;

while(p!=NULL)

{

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

p=p->next;

}

p=run;

while(p!=NULL)

{

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

p=p->next;

}

}

void InsertPrio(PCB \*in)

{

PCB \*fst,\*nxt;

fst=nxt=ready;

if(ready==NULL)

{

in->next=ready;

ready=in;

}

else

{

if(in->prio>=fst->prio)

{

in->next=ready;

ready=in;

}

else

{

while(fst->next!=NULL)

{

nxt=fst;

fst=fst->next;

}

if(fst->next==NULL)

{

in->next=fst->next;

fst->next=in;

}

else

{

nxt=in;

in->next=fst;

}

}

}

}

void InsertTime(PCB \*in)

{

PCB \*fst;

fst=ready;

if(ready==NULL)

{

in->next=ready;

ready=in;

}

else

{

while(fst->next!=NULL)

{

fst=fst->next;

}

in->next=fst->next;

fst->next=in;

}

}

void InsertFinish(PCB \*in)

{

PCB \*fst;

fst=finish;

if(finish==NULL)

{

in->next=finish;

finish=in;

}

else

{

while(fst->next!=NULL)

{

fst=fst->next;

}

in->next=fst->next;

fst->next=in;

}

}

void PrioCreate()//创建进程的过程

{

PCB \*tmp;

int i;

cout<<"输入进程的名字和进程所需时间："<<endl;

for(i=0;i<num;i++)//num为进程的数目

{

if((tmp=(PCB \*)malloc(sizeof(PCB)))==NULL)

{

perror("malloc");

exit(1);

}

cin>>tmp->name;//输入进程的名字

getchar();

cin>>tmp->needtime;//输入进程运行的时间

tmp->cputime=0;

tmp->state='W';

tmp->prio=50-tmp->needtime;

tmp->round=0;

tmp->count=0;

InsertPrio(tmp);

}

}

void TimeCreate()

{

PCB \*tmp;

int i;

cout<<"输入进程名字和进程时间片所需时间：";

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

{

if((tmp=(PCB \*)malloc(sizeof(PCB)))==NULL)

{

perror("malloc");

exit(1);

}

cin>>tmp->name;

getchar();

cin>>tmp->needtime;

tmp->cputime=0;

tmp->state='W';

tmp->prio=0;

tmp->round=2;

tmp->count=0;

InsertTime(tmp);

}

}

void Priority()

{

int flag=1;

GetFirst();

while(run!=NULL)

{

Output();

while(flag)

{

run->prio-=3;

run->cputime++;

run->needtime--;

if(run->needtime==0)

{

run->state='P';

run->count++;

InsertFinish(run);

flag=0;

}

else

{

run->state='W';

run->count++;

InsertTime(run);

flag=0;

}

}

flag=1;

GetFirst();

}

}

void RoundRun()

{

int flag=1;

GetFirst();

while(run!=NULL)

{

Output();

while(flag)

{

run->count++;

run->cputime++;

run->needtime--;

if(run->needtime==0)

{

run->state='F';

InsertFinish(run);

flag=0;

}

else if(run->count==run->round)

{

run->state='W';

run->count=0;

InsertTime(run);

flag=0;

}

}

flag=1;

GetFirst();

}

}