**郑州轻工业大学**

**《操作系统》课程****实验报告**

实验名称：短作业优先调度算法

姓 名： 原彬贺

学 号： 542001020223

地 点： 实验楼302

指导老师： 吴庆岗

专业班级： 计算机科学与技术20-02班

实验成绩：

1. 实验目的：

编写程序，实现SJF作业调度算法，模拟作业调度过程，加深对作业调度的理解。

二、实验内容：

实现SJF调度算法。实验要求：

1、设计作业控制块(JCB)的数据结构

–应包含实验必须的数据项，如作业ID、需要的服务时间、进入系统时间、完成时间，以及实验者认为有必要的其他数据项。

2、实现排序算法（将作业排队）

–策略1：按“进入系统时间”对作业队列排序(FCFS)

–策略2：按“需要的服务时间”对作业队列排序（SJF）

3、实现调度过程模拟

（1）每个作业用一个JCB表示，按步骤2方法作业排队，

（2）选择队首的作业，将其从后备队列移出

（3）（作业运行过程，在本实验中，无需实现，可认为后备队列上的作业一但被调度程序选出，就顺利运行完毕，可以进入第4步）

（4）计算选中作业的周转时间

（5）进行下一次调度（去往第2步）

4、实现结果输出

– 输出作业状态表，展示调度过程

* + 初始作业状态（未调度时）
  + 每次调度后的作业状态

三、实验代码

#include "stdio.h"

#include<iostream>

#include <vector>

#include <ctime>

#include <list>

#include <algorithm>

#include <iterator>

using namespace std;

#define FLT\_MAX 3.402823466e+38F

#define FLT\_MIN 1.175494351e-38F

class JCB {

private:

int id;

int startTime;

int doneTIme;

int serverTime;

int creatTime;

public:

JCB(int ID, int ServerTime,int CreattTiem) :id(ID), serverTime(ServerTime), creatTime(CreattTiem) {

}

int getid()

{

return id;

}

void setStartTime(int time)

{

startTime = time;

}

int getcreatTiem()

{

return creatTime;

}

int getstartTiem()

{

return startTime;

}

int getdoneTime()

{

return doneTIme;

}

int getserverTime()

{

return serverTime;

}

bool setdoneTime()

{

doneTIme = std::time(0);

}

bool cmpcreat(JCB const\* peer)

{

if (this->creatTime <= peer->creatTime)

{

return true;

}

else return false;

}

bool cmpserver(JCB const\* peer)

{

if (this->serverTime <= peer->serverTime)

{

return true;

}

else

{

return false;

}

}

};

class JCBworks {

private:

list<JCB\*> jcbworks;

int status;

public:

JCBworks(int stuts) :status(stuts)

{

}

void setStatus(int sta)

{

status = sta;

return;

}

void show()

{

int j = 0;

list<JCB\*>::iterator itr = jcbworks.begin();

int t = 0;

for (;itr != jcbworks.end();itr++)

{

if (status == 0)

{

cout << "当前时间： " << t << " " ;

cout << "第" << j << "个 id = " << (\*itr)->getid() << " starttime:= " << (\*itr)->getcreatTiem() << " srevertime:= " << (\*itr)->getserverTime() << endl;

t += (\*itr)->getserverTime();

}

else

{

cout << "当前时间： " << t << " " ;

cout << "第" << j << "个 id = " << (\*itr)->getid() << " starttime:= " << (\*itr)->getcreatTiem() << " srevertime:= " << (\*itr)->getserverTime() << endl;

t += (\*itr)->getserverTime();

}

j++;

}

cout << "全部完成，总花费：" << t << endl;

}

void appendJCB(JCB\* jcb)

{

auto it = jcbworks.begin();

switch (status)

{

case 0: {

for (; it != jcbworks.end(); it++)

{

if (jcb->cmpcreat(\*(it)))

{

jcbworks.insert(it, jcb);

break;

}

}

if (it == jcbworks.end())

{

jcbworks.insert(it,jcb);

}

break;

}

case 1:

{

for (; it != jcbworks.end(); it++)

{

if (jcb->cmpserver((\*it)))

{

jcbworks.insert(it, jcb);

break;

}

}

if (it == jcbworks.end())

{

jcbworks.insert(it, jcb);

}

break;

}

}

}

};

int main()

{

int nums[10] = { 11,23,14,12,15,12,16,17,13,17 };

int time[10] = { 10,3,1,5,6,1,4,6,2,7 };

JCBworks\* works = new JCBworks(0);

for (int i = 0; i < 10; i++)

{

JCB\* jcb = new JCB(i, nums[i],time[i]);

works->appendJCB(jcb);

}

cout << "creatTime" << endl;

works->show();

JCBworks\* works2 = new JCBworks(1);

for (int i = 0; i < 10; i++)

{

JCB\* jcb = new JCB(i, nums[i], time[i]);

works2->appendJCB(jcb);

}

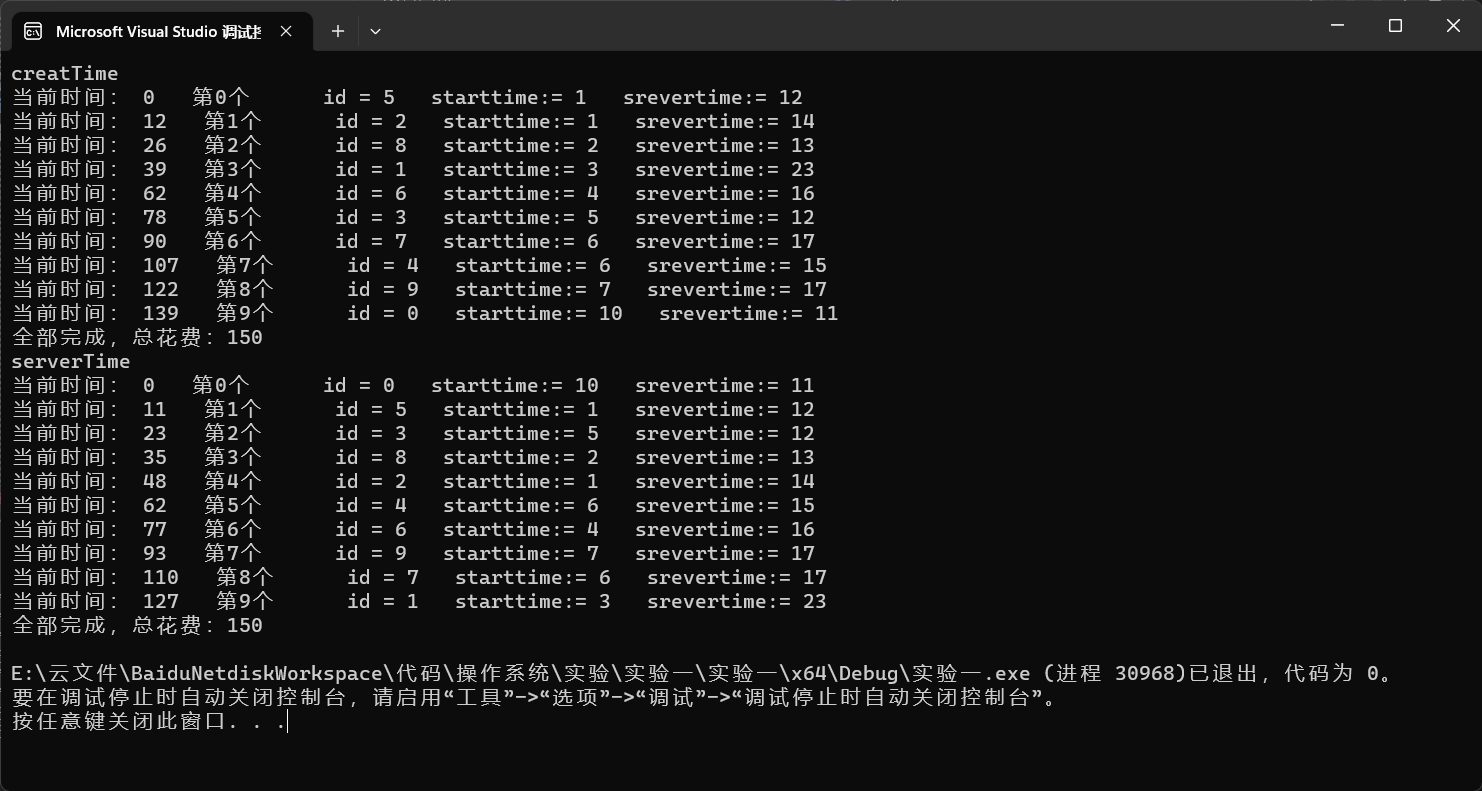
cout << "serverTime" << endl;

works2->show();

return 0;

}

四、实验结果



五、实验总结

本次实验我通过具体实现先来先服务调度算法和短作业优先调度算法，来进行任务的调度。通过本次自己实现该算法，使我对其原理以及具体的运行过程有了更加深刻和具体的认识，同时这两种算法的学习，也使我了解到更多进程调度的算法，这些算法使我的眼界更加开拓，给我一个全新的角度来看待问题。这次实验给我帮助很大，使我收获颇丰，使我将理论与实际牢牢结合，增强了我的动手能力与代码编写能力。

|  |
| --- |
| **评语**（不要删除，指导老师填写评语，放置在实验报告后面单独成页） |
|  |