**天津财经大学理工学院信息科学与技术系**

**《操作系统》实验报告**

实验 1 处理器调度算法

**班 级：软件2201**

**学 号：2022113179**

**姓 名：钟奇林**

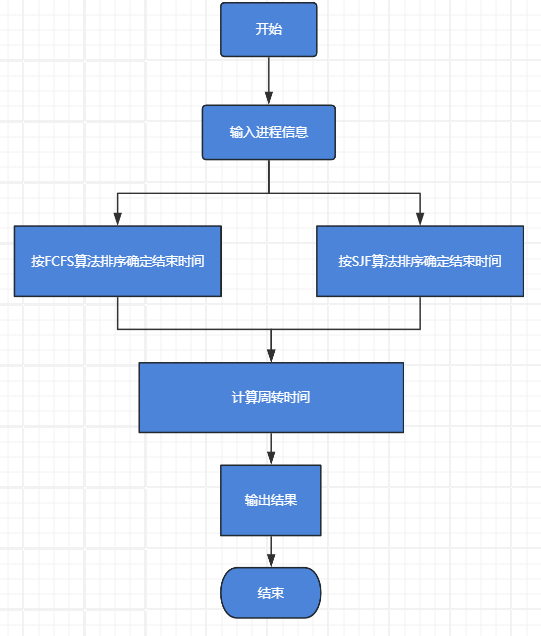
**一、实验目的**

通过本实验模拟实现操作系统对处理器调度方法，从而加深对处理器调度算法的理解。

**二、实验内容**

编写实现处理器调度的程序。要求实现先进先出（FIFO）调度算法或最短作业优先（SJF）。

各作业的作业号、提交时间和执行时间如下所示，按照先进先出调度算法或最短作业优先调度算法编写程序，输出调度作业序列。

**三、实验结果**

1、先来先服务(FCFO)算法

核心代码说明如下：

// 先来先服务调度算法

void SchedulingAlgorithm::FIFO(Process\* p\_a, int n) {

sort(p\_a, p\_a + n, [](Process x, Process y)->bool {return x.arrival\_time < y.arrival\_time; }); // 按到达时间排序

this->cpu\_run\_time = p\_a[0].arrival\_time; // 初始CPU运行时间为第一个进程的到达时间

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

if (p\_a[i].arrival\_time > this->cpu\_run\_time) { // 到达时间超过当前cpu运行时间

this->cpu\_run\_time = p\_a[i].arrival\_time; // 更新CPU运行时间

}

p\_a[i].start\_time = this->cpu\_run\_time; // 设置开始时间

this->cpu\_run\_time += p\_a[i].expected\_run\_time; // 更新CPU运行时间

p\_a[i].end\_time = this->cpu\_run\_time; // 设置结束时间

}

}

// 最短作业优先调度算法

void SchedulingAlgorithm::SJF(Process\* p\_a, int n) {

sort(p\_a, p\_a + n, [](Process x, Process y)->bool {return x.arrival\_time < y.arrival\_time; }); // 按到达时间排序

this->cpu\_run\_time = p\_a[0].arrival\_time; // 初始CPU运行时间为第一个进程的到达时间

int i = 0;

while (i < n) {

p\_a[i].start\_time = this->cpu\_run\_time; // 设置开始时间

this->cpu\_run\_time += p\_a[i].expected\_run\_time; // 更新CPU运行时间

p\_a[i].end\_time = this->cpu\_run\_time; // 设置结束时间

// 更新

++i;

int amount = 0;

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

if (p\_a[j].arrival\_time <= this->cpu\_run\_time)amount++;

sort(p\_a + i, p\_a + i + amount, [](Process x, Process y)->bool {return x.expected\_run\_time < y.expected\_run\_time; }); // 重新排序

//printProcessTable(p\_a, n);

}

}

运行结果如下：



**四、实验总结**

这次试验是模拟操作系统如何对作业进行调度，从后备队列中选取某些作业调入内存。使用的是短作业优先调度算法，整体实现这一算法是比较容易的