**实验三 进程管理**

**一、实验目的及要求**

**实验目的：**

1. 验证并解释进程的基本概念。

2. 验证并解释进程管理的基本原则。

**实验类型：**验证性实验  **实验学时：**4学时 **实验人数：**1人

**二、实验内容**

1.进程的概念。

要求生成两个进程，启动进程，通过任务管理查看进程。

2.利用先来先服务算法模拟进程调度的过程，理解进程管理的基本原则。

要求：

（1）理解先来先服务算法的概念。

（2）使用类和列表模拟进程调度的过程。

**三、实验条件**

Windows 7操作系统，Python 3.7。

**四、实验相关知识点**

进程、进程调度、先来先服务算法。

进程调度算法：采用先来先服务、短作业、动态最高优先数优先的调度算法（即把处理机分配给优先数最高的进程）。

先来先服务调度算法—— FCFS法（First Come First Serve）

• 基本思想：按照进程变为就绪状态的先后次序，调入系统或分派CPU，换句话说，调度程序每次选择的进程是等待时间最久的，而不管其运行时间的长短。

• 特点

– 系统开销小，实现简单

– 比较有利于长作业和CPU繁忙的作业，而不利于短作业和

I/O繁忙的作业。

• 在实际操作系统和一般应用程序中较常采用FCFS算法，且

通常和其他算法配合起来。

每个进程有一个进程控制块（ PCB）表示。进程控制块可以包含如下信息：

进程名—进程标示数 ID

到达时间—进程的到达时间为进程输入的时间。

进程还需要运行时间ALLTIME，进程运行完毕ALLTIME=0，

已用CPU时间----CPUTIME、

进程状态—STATE

周转时间=作业完成时间−作业提交时间

平均周转时间=（作业1的周转时间+...+作业n的周转时间n平均周转时间=作业1的周转时间+...+作业n的周转时间）/n

带权周转时间=作业周转时间/作业实际运行时间

平均带权周转时间=（作业1的带权周转时间+...+作业n的带权周转时间）/n

**五、实验步骤[或实验参考代码]**

1.

import os

import time

from multiprocessing import Process # 进程模块

def func():

time.sleep(100)

while(1):

print('in func',os.getpid(),os.getppid())

if \_\_name\_\_ == '\_\_main\_\_':

print('in main',os.getpid(),os.getppid())

p1 = Process(target=func) # 进程对象

p1.start() # 向操作系统提交了一个开启子进程的申请

print(p1)

p2 = Process(target=func) # 进程对象

p2.start() # 向操作系统提交了一个开启子进程的申请

print(p2)

print('主进程 的 代码执行结束了')

# 原理

# if \_\_name\_\_ == '\_\_main\_\_':

# 使用python都是调用操作系统的命令来启动进程

# 同样使用python 不同的操作系统的操作是不同的

# 对于windows来说 必要加if \_\_name\_\_ == '\_\_main\_\_':

# 对于linux ios来说 不必要加if \_\_name\_\_ == '\_\_main\_\_':

2.

import random

class PCB:

def \_\_init\_\_(self,pid,arr\_time,all\_time,cpu\_time,state): ##初始化进程

self.pid=pid

self.arr\_time=arr\_time

self.all\_time=all\_time

self.cpu\_time=cpu\_time

self.state=state

def Output(self): ##sjf fcfs输出

print("进程"+str(self.pid),"正在执行，到达时间:"+str(self.arr\_time),

"还需运行时间:"+str(self.all\_time),"已运行时间:"+str(self.cpu\_time))

def toRun(self): ##将状态置为Run

self.state="Run"

def toFinish(self): ##将状态置为Finish

self.state="Finish"

def toReady(self): ##将状态置为Ready

self.state="Ready"

def running(self): ##进程运行时状态变化

self.all\_time-=1

self.cpu\_time+=1

def init(num):##初始化进程，生成四个进程并按到达时间将它们放入列表list1

list1=[]

for i in range(num):

list1.append(PCB(str(i),random.randint(10,15),

random.randint(1,10),0,"Ready"))

for i in range(len(list1)-1):

for j in range(i+1,len(list1)):

if list1[i].arr\_time>list1[j].arr\_time:

list1[i],list1[j]=list1[j],list1[i]

return list1

def fcfs(list1):##先来先服务

time=0

while 1:

print("time:",time)

if time>=list1[0].arr\_time:

list1[0].running()

list1[0].Output()

if list1[0].all\_time==0:

print("进程"+list1[0].pid+"执行完毕,周转时间："+str(time-list1[0].arr\_time+1)+"\n")

list1.remove(list1[0])

time+=1

if not list1:

break

if \_\_name\_\_=="\_\_main\_\_":

while 1:

n=input("请选择算法(1、先来先服务):")

if n=="1":

list1=init(4)

for i in list1:

i.Output()

fcfs(list1)

else:

print("输入错误，请重新输入！")