实 验 报 告

课程名称：操作系统试验

实 验 二：进程调度

班　　级：2020 级计算机科学与技术

学生姓名：胡诚皓

学　　号：20201060330

专　　业：2020 级计算机科学与技术

指导教师：杨旭涛

学　　期：2022－2023 学年秋季学期

成　　绩：

云南大学信息学院

一、实验目的

1、熟悉进程的定义和描述，熟悉进程控制块；

2、掌握进程的状态定义及其转换过程；

3、掌握进程的基本调度算法，包括先来先服务，轮转法，优先级法，多级反馈轮转法，最短进程优先法，最高响应比优先法等；

二、知识要点

1、进程控制块 PCB；

2、进程的初始化、就绪、执行、等待和终止状态；

3、先来先服务，轮转法，优先级法，多级反馈轮转法，最短进程优先法，最高响应比优先法等调度算法；

三、实验预习（要求做实验前完成）

1、了解 linux 系统中常用命令的使用方法；

2、掌握进程 PCB 控制块的内容和描述；

3、掌握系统状态的转换过程；

4、掌握常用进程调度算法的原理

四、实验内容和试验结果

结合课程所讲授内容以及课件中的试验讲解，完成以下试验。请分别对试验过程和观察到的情况做描述和总结，并将试验结果截图附后。

模拟程序设计逻辑

本篇报告中的进程调度模拟均使用 Python 完成。下面先说明一下编写的 PCB 类与 PCB 链表类 PCBLinkedList。

PCB 类中的 5 个静态变量，用于统一表示某个 PCB 当前的状态，在本次实验中，并没有全部用到。但考虑到程序的完整性以及可扩展性，仍然将进程的各个可能状态设计在其中，如下表所示：

变量名	值	说明
state_create	CREATE	进程刚被创建
state_running	RUNNING	进程正在 CPU 上运行
state_ready	READY	进程处于就绪态
state_waiting	WAITING	进程处于阻塞态
state_done	TERMINATED	进程已经完成

对于每一个 PCB 对象，其属性包含进程标识号 PID、需要运行的时间（即预计运行时间)total_time、已经运行的时间 cpu_time、进程创建时间 create_time、进程优先级 priority

对于每一个 PCBLinkedList 对象，其属性包含用于存放就绪或正在运行的 PCB 对象的列表 list、用于存放已经运行完成的 PCB 对象的列表 trash、列表 list 中的各 PCB 对象的 PID 集合 pids。

PCBLinkedList 类中提供了 PCB 链表需要提供的功能，往链表中添加一个或多个新进程 add_one 或 add_many、判断就绪队列或正在运行的进程中是否已使用某 PID 号 pid_exist、移除已完成的进程 delete_one、判断 PCB 链表中是否有元素 is_empty、获取 PCB 链表的头尾元素 get_head 与 get_tail。

另外，还抽象出来了两个工具方法 init_pcbs 与 user_interact，分别用于让用户往 PCB 表中添加进程与每一次 CPU 时间片结束之后的用户交互。

1、通过编程模拟实现轮转法进程调度算法。

系统中的每个进程用一个进程控制块 PCB 表示；将多个进程按输入顺序排成就绪队列链表（进程信息从键盘录入）；按进程在链表中的顺序依次调度，每个被调度的进程执行一个时间片，然后回到就绪队列，“已运行时间”加 1；若进程“要求运行时间”==“已运行时间”，则将其状态置为“结束”，并退出队列；运行程序，显示每次调度时被调度运行的进程 id，以及各进程控制块的动态变化过程。

PPT 中的代码是用 C 完成的,此处使用上述自己编写的 PCB 与 PCBLinkedList 进行重现。由于 Python 中并没有“指针”这一概念，对于 PCB 链表，使用迭代器完成轮询。当迭代器为最后一个元素时,重新获取迭代器即可。算法描述如下：

① 若迭代器不存在或迭代器已经到达结尾,则获取一个 PCB 链表的迭代器。

② 从迭代器中获取一个 PCB，将其运行一个 CPU 时刻

③ 若该进程已经执行完成，即已运行时间与预计运行时间相等，则将其从 PCB 链表中移除

④ 若用户不需要继续进行交互，则直接输出当前 PCB 链表情况

⑤ 进入用户交互逻辑

下图的例子中，一开始先创建了 3 个进程 1、2、3，需要执行的时间分别为 5、3、6 个 CPU 时间刻，在 CPU 运行过 2 个时间刻后，添加一个进程，PID 为 4，需要执行的时间为 2。模拟情况如下图所示：

```
要创建的进程数量：3
输入第1个进程的pid、预计执行时间、优先级（可缺省，默认为5），以空格分隔
1 5
输入第2个进程的pid、预计执行时间、优先级（可缺省，默认为5），以空格分隔
2 3
输入第3个进程的pid、预计执行时间、优先级（可缺省，默认为5），以空格分隔
3 6
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         1/5                     5
2       CREATED         0/3                     5
3       CREATED         0/6                     5

当前CPU时间为1，是否添加进程(y/N)
是否继续运行到结束(y/N)
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         1/5                     5
2       CREATED         1/3                     5
3       CREATED         0/6                     5

当前CPU时间为2，是否添加进程(y/N)y
要创建的进程数量：1
输入第1个进程的pid、预计执行时间、优先级（可缺省，默认为5），以空格分隔
4 2
```

```
4 2
是否继续运行到结束(y/N)
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         1/5                     5
2       CREATED         1/3                     5
3       CREATED         1/6                     5
4       CREATED         0/2                     5

当前CPU时间为3，是否添加进程(y/N)
是否继续运行到结束(y/N)y
当前CPU时间为8，4号进程完成
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         2/5                     5
2       CREATED         2/3                     5
3       CREATED         2/6                     5
4       TERMINATED      2/2                     5
```

```
当前CPU时间为10，2号进程完成
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         3/5                     5
3       CREATED         2/6                     5
4       TERMINATED      2/2                     5
2       TERMINATED      3/3                     5

当前CPU时间为13，1号进程完成
------------------------------
PID     Status          Run_time/Need_time      Priority
3       CREATED         3/6                     5
4       TERMINATED      2/2                     5
2       TERMINATED      3/3                     5
1       TERMINATED      5/5                     5

当前CPU时间为16，3号进程完成
------------------------------
PID     Status          Run_time/Need_time      Priority
4       TERMINATED      2/2                     5
2       TERMINATED      3/3                     5
1       TERMINATED      5/5                     5
3       TERMINATED      6/6                     5
```

2、参考第一题的描述，通过编程模拟实现动态优先级轮转调度算法。

与前面的单纯的轮转调度法不同，动态优先级轮转调度加入了优先级的机制，并且这个优先级会随着时间的推移逐渐变化。此处约定优先级用整数表示，且数值越大，优先级越高。每次调度时，都会选择优先级最高的进行放到 CPU 上运行。没运行一次，对应进程的优先级都会减一。此处为了防止优先级过低导致的一些不可预见的问题，约定优先级最低为-50。若进程的优先级减到-50 仍未运行完成，也不再往下减。下面是算法描述：

① 若用于选取优先级最高的全局优先队列不存在，则创建之，并把各个要运行的进程放入其中。

② 取出优先级最高的进程，运行一个 CPU 刻，并将其优先级减一（保证不低于-50）。

③ 若该进程已经执行完成，即已运行时间与预计运行时间相等，则将其从 PCB 链表中移除；否则，就将这个 PCB 重新放入优先队列中。

④ 若用户不需要继续进行交互，则直接输出当前 PCB 链表情况。

⑤ 进入用户交互逻辑。

作为对比，和前面的轮转法调度使用相同的例子，分别给予 1、2、3 号进程 8、3、10 的优先级，给予在 CPU 时间为 2 时加入的 4 号进程 12 的优先级。模拟运行结果如下图所示：

```
要创建的进程数量：3
输入第1个进程的pid、预计执行时间、优先级（可缺省，默认为5），以空格分隔
1 5 8
输入第2个进程的pid、预计执行时间、优先级（可缺省，默认为5），以空格分隔
2 3 3
输入第3个进程的pid、预计执行时间、优先级（可缺省，默认为5），以空格分隔
3 6 10
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         0/5                     8
2       CREATED         0/3                     3
3       CREATED         1/6                     9

当前CPU时间为1，是否添加进程(y/N)
是否继续运行到结束(y/N)
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         0/5                     8
2       CREATED         0/3                     3
3       CREATED         2/6                     8

当前CPU时间为2，是否添加进程(y/N)y
要创建的进程数量：1
输入第1个进程的pid、预计执行时间、优先级（可缺省，默认为5），以空格分隔
4 2 12
是否继续运行到结束(y/N)y

当前CPU时间为4，4号进程完成
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         0/5                     8
2       CREATED         0/3                     3
3       CREATED         2/6                     8
4       TERMINATED      2/2                     10


当前CPU时间为12，3号进程完成
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         4/5                     4
2       CREATED         0/3                     3
4       TERMINATED      2/2                     10
3       TERMINATED      6/6                     4
```

```
当前CPU时间为13，1号进程完成
------------------------------
PID     Status          Run_time/Need_time      Priority
2       CREATED         0/3                     3
4       TERMINATED      2/2                     10
3       TERMINATED      6/6                     4
1       TERMINATED      5/5                     3


当前CPU时间为16，2号进程完成
------------------------------
PID     Status          Run_time/Need_time      Priority
4       TERMINATED      2/2                     10
3       TERMINATED      6/6                     4
1       TERMINATED      5/5                     3
2       TERMINATED      3/3                     0
```

3、参考第一题的描述，通过编程模拟实现最高响应比优先进程调度算法。

响应比定义为$R=\frac{W+S}{S}=1+\frac{W}{S}$，其中$W$为进程创建后已经等待了的时间，$S$为预计执行时间，即进程需要执行的时间。可以发现，随着进程等待时间的增长，其响应比会越来越高；并且在等待时间相同的情况下，需要执行时间越短的响应比越高，对长短进程都有所兼顾。在实现过程中，响应比其实可以直接视为进程的优先级。

由于每次运行完成后需要对每个进程的响应比作出更新，因而可以在这个遍历中直接对比得到响应比最高的进程，无需再使用优先队列浪费时间。另外，不需要使用额外的属性记录等待时间，等待时间事实上等于“当前 CPU 时间-进程创建时间-进程已运行时间”。具体算法描述如下：

① 遍历 PCB 链表更新响应比，同时找出更新后响应比最大的进程。

② 将响应比最高的进程运行一个 CPU 时间刻。

③ 若当前进程运行完成，将其从 PCB 表中移除。

④ 若用户不需要继续进行交互，则直接输出当前 PCB 链表情况。

⑤ 进入用户交互逻辑。

同样使用与第一题中相同的进程进行模拟以便于对比模拟结果。模拟运行结果如下图：

```
要创建的进程数量：3
输入第1个进程的pid、预计执行时间、优先级（可缺省，默认为5），以空格分隔
1 5
输入第2个进程的pid、预计执行时间、优先级（可缺省，默认为5），以空格分隔
2 3
输入第3个进程的pid、预计执行时间、优先级（可缺省，默认为5），以空格分隔
3 6
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         0/5                     1.2
2       CREATED         1/3                     1.3333333333333333
3       CREATED         0/6                     1.1666666666666667

当前CPU时间为1，是否添加进程(y/N)
是否继续运行到结束(y/N)
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         1/5                     1.4
2       CREATED         1/3                     1.3333333333333333
3       CREATED         0/6                     1.3333333333333333

当前CPU时间为2，是否添加进程(y/N)y
要创建的进程数量：1
输入第1个进程的pid、预计执行时间、优先级（可缺省，默认为5），以空格分隔
4 2
```

```
是否继续运行到结束(y/N)y

当前CPU时间为5，2号进程完成
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         1/5                     1.8
3       CREATED         0/6                     1.8333333333333335
4       CREATED         1/2                     2.0
2       TERMINATED      3/3                     2.0


当前CPU时间为6，4号进程完成
------------------------------
PID     Status          Run_time/Need_time      Priority
1       CREATED         1/5                     2.0
3       CREATED         0/6                     2.0
2       TERMINATED      3/3                     2.0
4       TERMINATED      2/2                     2.5
```

```
当前CPU时间为13，1号进程完成
------------------------------
PID     Status          Run_time/Need_time      Priority
3       CREATED         3/6                     2.666666666666667
2       TERMINATED      3/3                     2.0
4       TERMINATED      2/2                     2.5
1       TERMINATED      5/5                     2.8

当前CPU时间为16，3号进程完成
------------------------------
PID     Status          Run_time/Need_time      Priority
2       TERMINATED      3/3                     2.0
4       TERMINATED      2/2                     2.5
1       TERMINATED      5/5                     2.8
3       TERMINATED      6/6                     2.833333333333333
```

下面是以 word 附件形式附加的 Python 源代码，双击可以打开。若被 Office 阻止访问无法打开，可以选中后进行复制，粘贴到任一文件夹中即可。

exp2_1.py