实验5 进程调度模拟

实验任务

- 基于离散事件仿真模拟进程调度
- 输入：一组作业，包括

  （作业名，到达时间，所需运行时间，优先级）
- 输出：每个作业的到达时间、开始运行时间、结束时间

离散事件仿真

- 将系统中发生的动作都作为事件(Event)处理
- 离散事件仿真器就是按事件发生的顺序逐个处理事件，在处理事件的过程中可能会插入新的事件

离散事件仿真器的框架

1. 系统时间
   - 用于存储当前的时钟，每次执行一个事件之前都将仿真器的系统时间推进到事件的发生时刻
2. 事件
   - 通常用一个类层次来表示不同的事件，而用Event来表示所有事件的基类
   - 每个事件至少应具有发生时刻、事件类型、必要的事件属性等信息
3. 一个用于存储事件的优先队列queue
   – 队列按照事件的发生时间排序
4. 队列初始化
   - 将初始的已知事件插入队列
5. 事件处理循环

```
while(queue is not empty){
    remove the first event from queue;
    process the event; //possibly adding new event(event to happen in the
                       //future) to queue;
}
```

进程调度的模拟

- 例如模拟SJF
  - 系统中存在两种事件：任务到达事件(ArrivalEvent)和任务运行结束事件(FinishEvent)
  - 每个任务到达是已知的，作为ArrivalEvent
  - 队列初始化即将ArrivalEvent加入优先队列
  - 事件处理循环从事件队列里面取出一个Event，判断其为ArrivalEvent还是FinishEvent，对不同的事件执行不同的处理方式

进程调度的模拟

- ArrivalEvent的处理
  - 看CPU是否被占用
    - 若占用，则将待处理任务添加到就绪队列
    - 若不被占用，则占据CPU运行，并且生成一个新的事件FinishEvent, 该事件的发生时刻为当前时间+作业的运行时间，即表示在未来的某个时刻作业会运行结束，请求系统处理该事件
    - 注：如果是抢占式调度，则可能会抢占已有作业运行
- FinishEvent的处理
  - 输出相关作业结束运行的信息
  - 从就绪队列中选择一个作业运行时间最短的作业，让其占据cpu，并且生成一个新的FinishEvent事件，该事件的发生时刻为当前时间+作业的运行时间

注：插入的事件是未来事件

第三类事件

- 时钟事件
  - 对于分时调度，还需要增加时钟TimerEvent
  - 当一个进程占据CPU运行时，
    - 如果进程的剩余执行时间小于时间片，则生成FinishEvent
    - 如果进程的剩余执行时间大于时间片，则生成一个TimerEvent, 插入事件队列，表示在未来某个时刻中断该进程的执行，并调度另一个进程执行

要求

- 选择一：
  - 实现FCFS, SJF，HRRF，RR，SRTF这些调度算法中的至少四种
- 选择二：
  - 实现多级反馈队列调度算法
    - 3个队列(第0,1,2队列)，第i个队列每次被分配的时间片为2^i

进程调度测试案例

Process	Arrival Time	Service Time
A	0	3
B	2	6
C	4	4
D	6	5
E	8	2