## 练习1: 使用 Round Robin 调度算法(不需要编码)

请理解并分析sched\_calss中各个函数指针的用法,并接合Round Robin 调度算法描

ucore的调度执行过程

共5个函数指针。

Init： 初始化调度算法

Enqueue：把一个进程放入运行队列中，会把该进程状态变为runable

Dequeue：从运行队列中把进程取出，并开始运行进程

Pick\_next：调用下一个可以运行的进程

Proc\_tick：用于时钟中断的调用，保证一个进程不会运行时间过长。

以上五个函数都封装在sched类中，调用的时候初始实例之后本身调用这五个函数，保证调度算法可以正常运行。

调度执行过程：

1， 查看代码发现调度算法的init函数仅存在在wakeup\_proc和schedule，前者的作用在于将某一个指定进程放入可执行进程队列中，后者在于将当前执行的进程放入可执行队列中，然后将队列中选择的下一个执行的进程取出执行；

2， 当需要将某一个进程加入就绪进程队列中，则需要将这个进程的能够使用的时间片进行初始化，然后将其插入到使用链表组织的队列的对尾；

3， 当需要将某一个进程从就绪队列中取出的时候，只需要将其直接删除即可；

4， 当需要取出执行的下一个进程的时候，只需要将就绪队列的队头取出即可；

5， 每当出现一个时钟中断，则会将当前执行的进程的剩余可执行时间减1，一旦减到了0，则将其标记为可以被调度的，这样就会调用schedule函数将这个进程切换出去；

请在实验报告中简要说明如何设计实现”多级反馈队列调度算法“,给出概要设计,鼓励给

出详细设计

在proc\_struct中添加总共N个多级反馈队列的入口，每个队列都有着各自的优先级，可以采用数组简化，编号越大的队列优先级约低，并且优先级越低的队列上时间片的长度越大，为其上一个优先级队列的两倍；并且在PCB中记录当前进程所处的队列的优先级；

处理调度算法初始化的时候需要同时对N个队列进行初始化；

在处理将进程加入到就绪进程集合的时候，观察这个进程的时间片有没有使用完，如果使用完了，就将所在队列的优先级调低，加入到优先级低1级的队列中去，如果没有使用完时间片，则加入到当前优先级的队列中去；

在同一个优先级的队列内使用时间片轮转算法；

在选择下一个执行的进程的时候，有限考虑高优先级的队列中是否存在任务，如果不存在才转而寻找较低优先级的队列；（有可能导致饥饿，为了避免优先级较低的进程出现饥饿现象，可以对每个优先级设置一定的选中概率，然后选出一个优先级，找到这个优先级中的第一个进程返回。）

从就绪进程集合中删除某一个进程就只需要在对应队列中删除即可；

处理时间中断的函数不需要改变和RR算法一样；

算法结束。