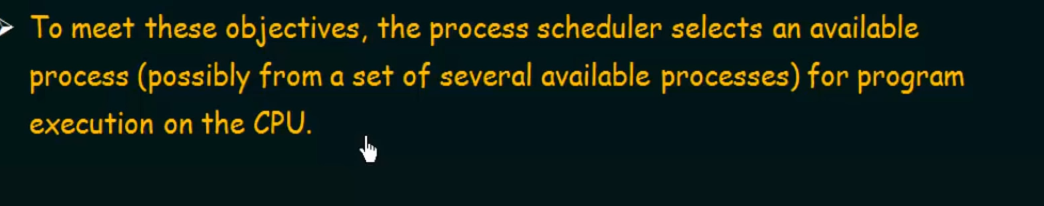
  
The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.

Process scheduling is an essential part of a Multiprogramming operating systems. Such operating systems allow more than one process to be loaded into the executable memory at a time and the loaded process shares the CPU using time multiplexing.

进程调度是进程管理器的活动，它处理从CPU中删除正在运行的进程以及根据特定策略选择另一个进程。进程调度是多程序设计操作系统的重要组成部分。这样的操作系统允许一次加载多个进程到可执行内存中，并且加载的进程使用时间复用共享CPU。



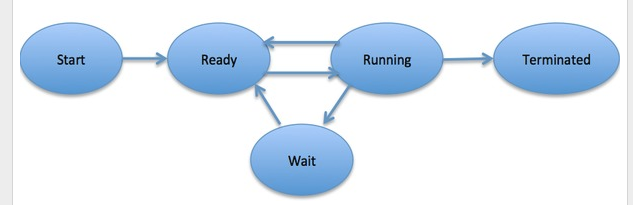
process manager handle process on a particular strategy

R p0 response time， 第一次在cpu中定位所需时间//什么时候开始运行

turnaround time"周转时间， process第一次进入ready state到process结束running state 的时间//ready state+running time ready time就是wait time

wait time wpi，process所需的所有时间来从非running状到ready状态，累加起来直到彻底结束 //所有wait time

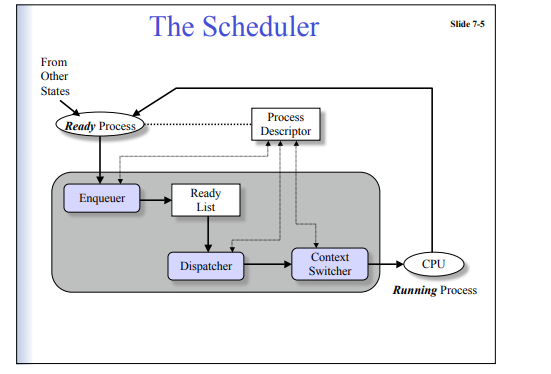
service time tpi 所需要的所有runninng time的和（在CPU中的time）//runing time

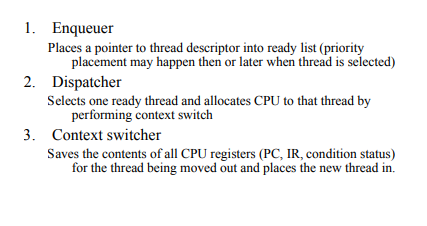


start 到ready就是response time，turn aroung time，，wait timeservice time

job queue:一旦process进入系统，那么他就是job queue

ready queue:在 main memory中的process,ready并等待execute



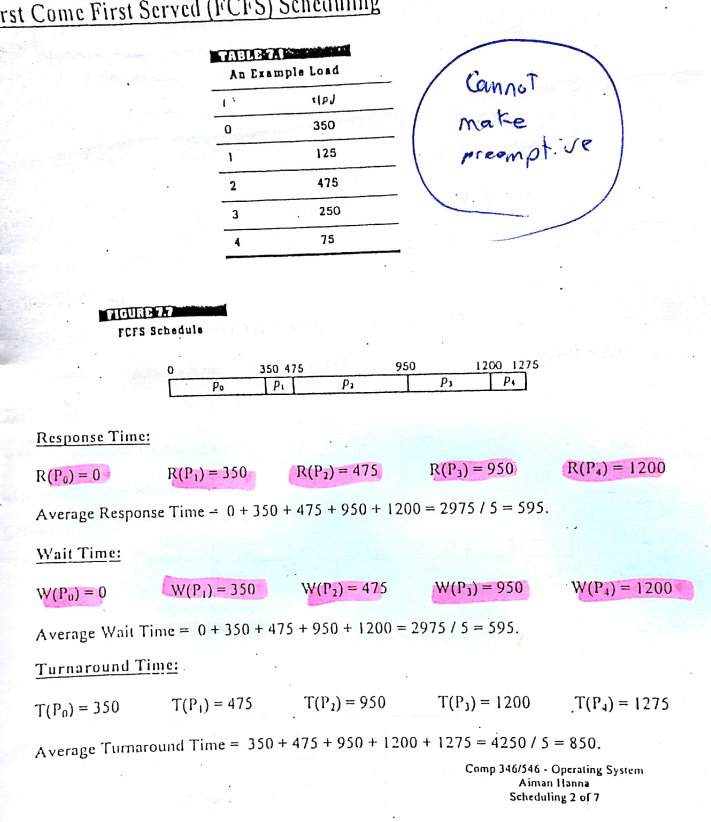


Non-preemptive scheduling: never wait a second time

first come first served sccheduling

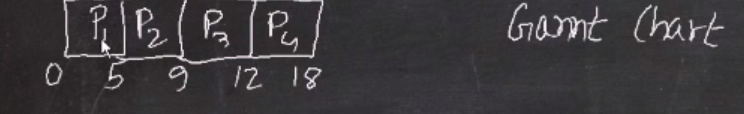
wait time=所有的相加除以数量

arrival time决定顺序



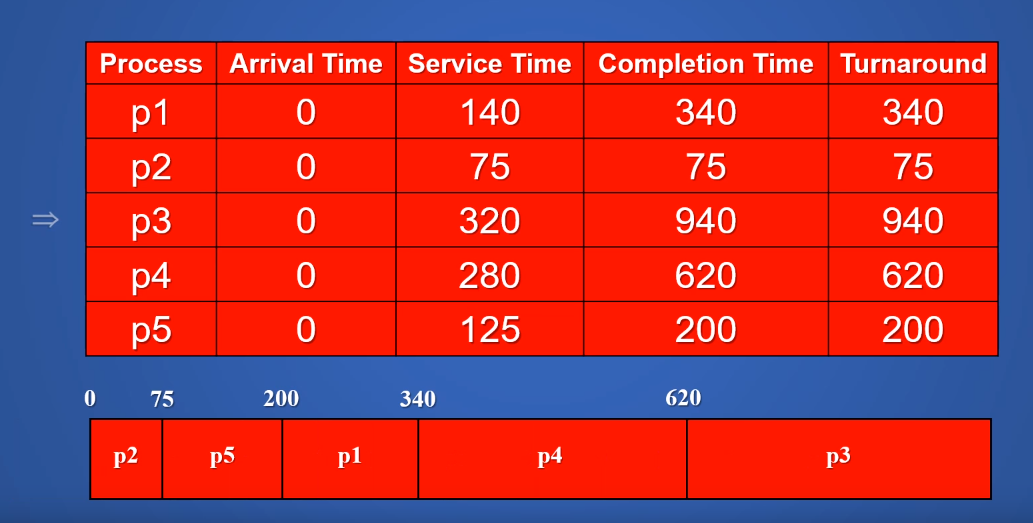
wait time是开始那个节点，turnaround time是结束后的那个节点

shortest job next，查看所有ready process，进行最短service time

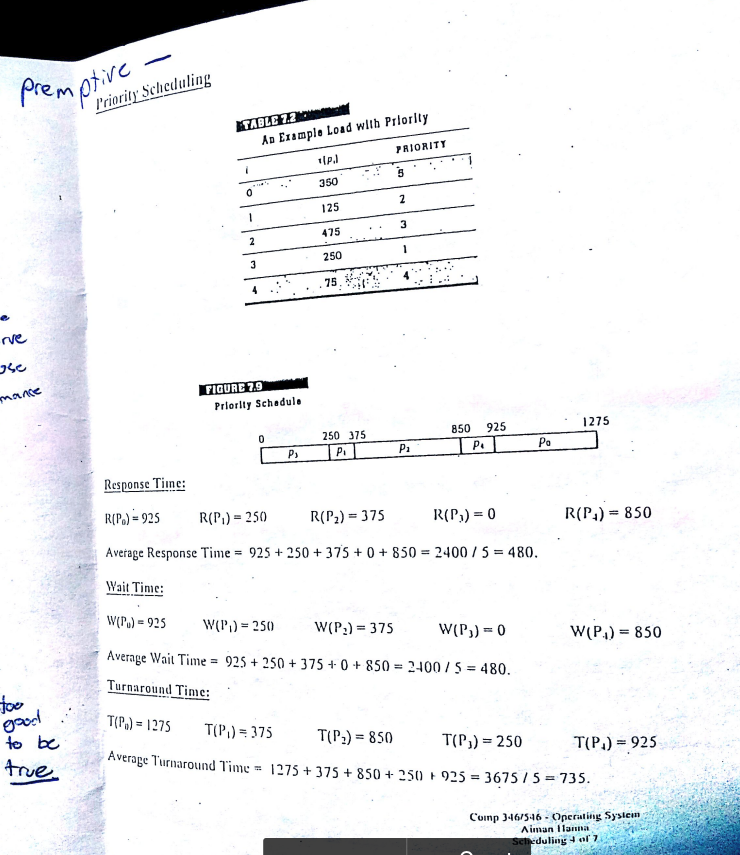


这个叫gannt chart

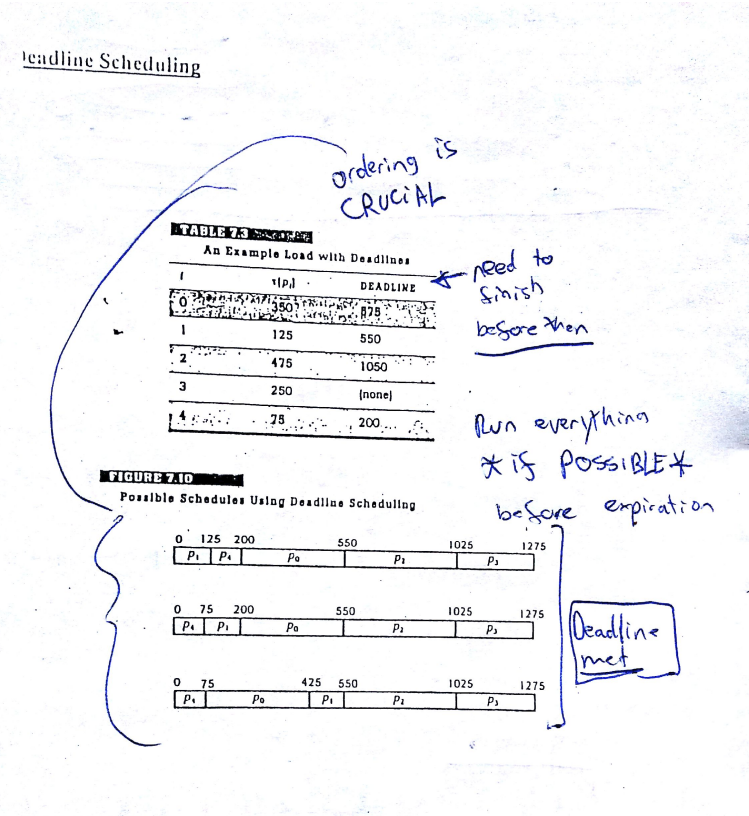
在复习一遍，进入cpu response time，长度叫做service time，一直到她开始是wait time//事实上是开始到进入时间，而这里arrival time全都默认为0了，一直到她结束是turnaround time



priority scheduling



每个都有优先级，按这个顺序排



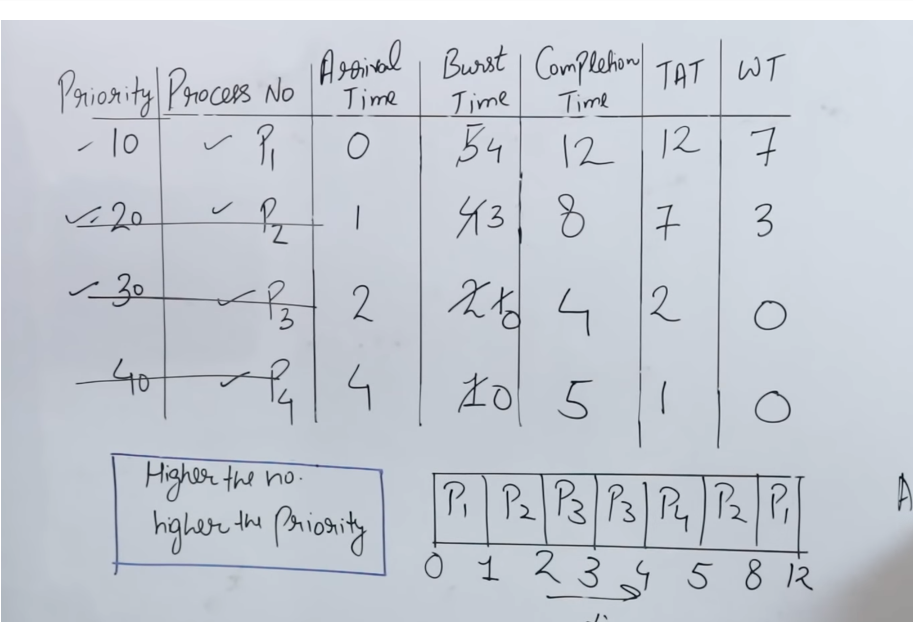
再deadline结束之前一定要完成

有时候不一定能结束dead line

non premptive Once resources(CPU Cycle) are allocated to a process, the process holds it till it completes its burst time or switches to waiting state.s，没有overhead

====  
premptive scheduling   
In this resources(CPU Cycle) are allocated to a process for a limited time.

Process can be interrupted in between.



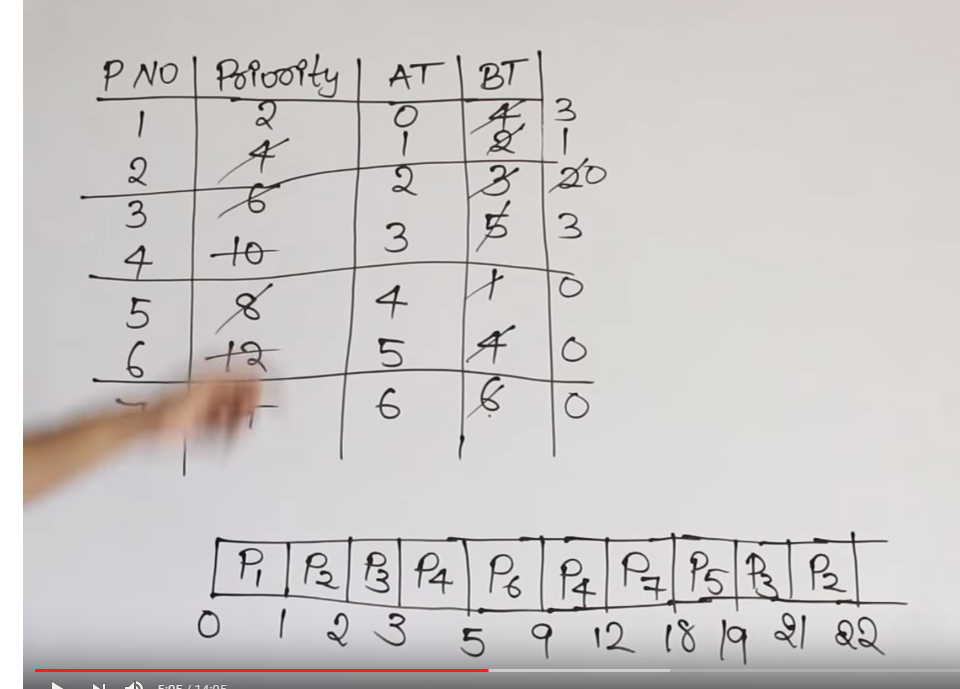
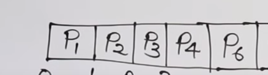
burst time是service time

arrival time是response time

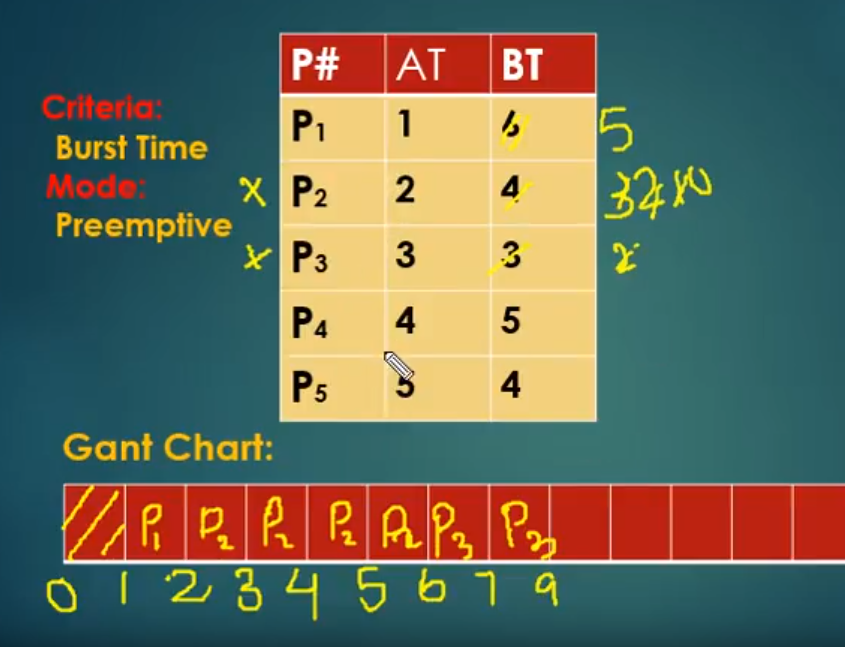
优先级更高arrival我们应该直接替换并减去对应service time

优先级更低就不管，

这一过程上升过程

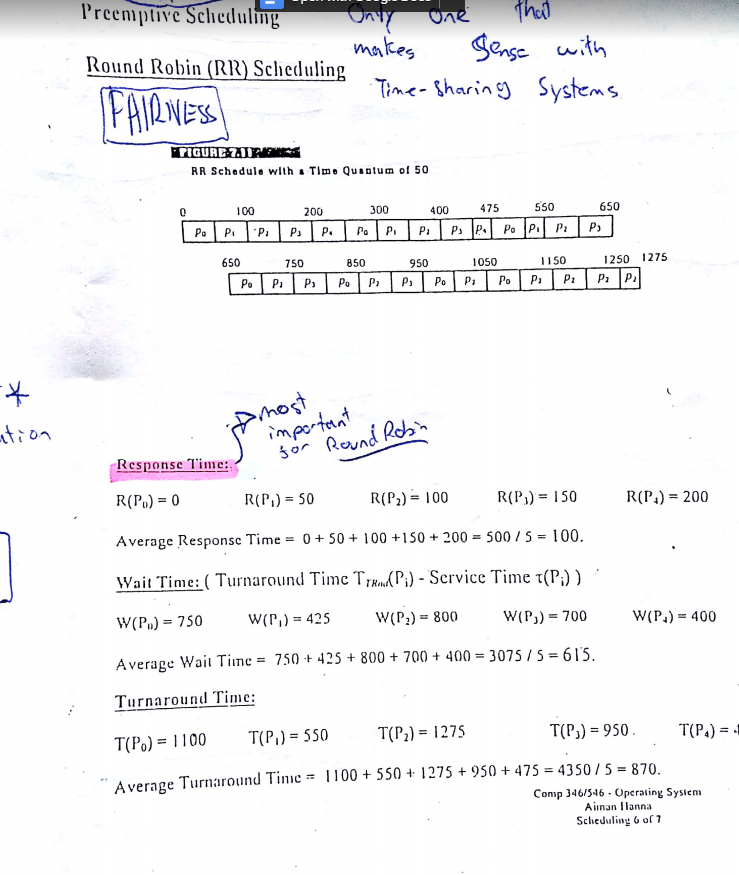
 

然后直接下降过程，按priority从高到小全部直接burst满



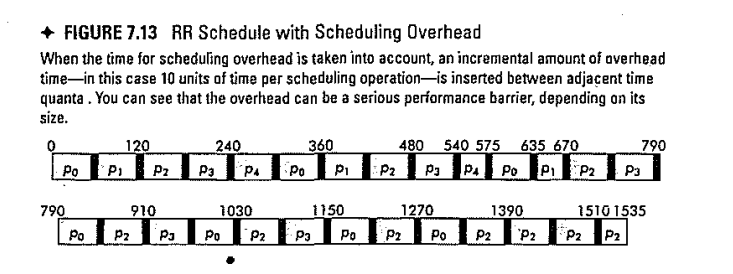
SJN 更简单， 同一时间选择·burst time最小的，注意AT，有的还没开始所以不参与比较

round robbin

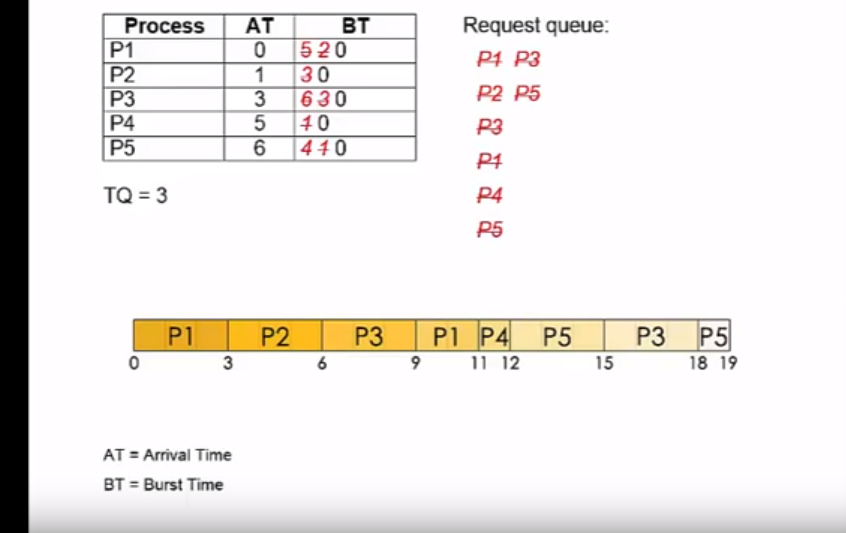


RR就是多久轮一次，wait time就是结束时间减去总运行时间，

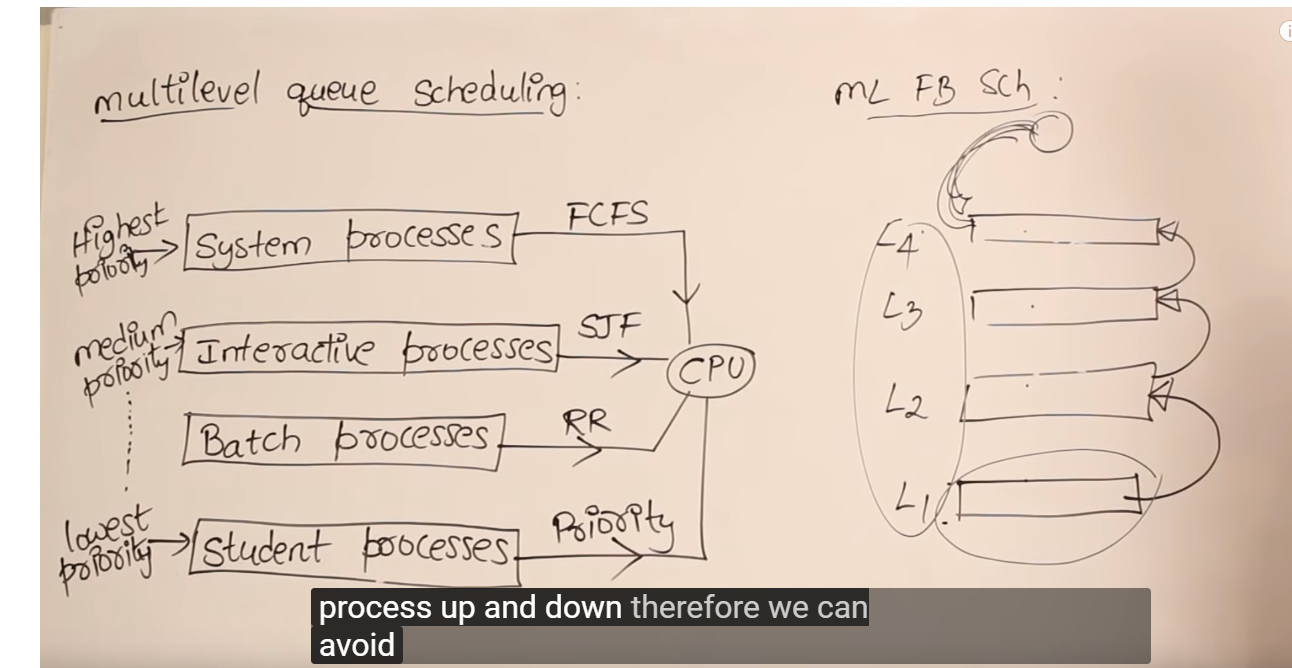
turnaround time就是最终结束时候，



考虑overhead的rr，quanty越小，效果越差



顺序问题·

.

multilevel queue scheduling

把不同类型的process给一个优先级，作为不同的ready queue，这一类的process没做完，永远不会下一类。

但是会产生starvation

用feedback，经过一定时间，改换queue的优先级，这样就不会有starvation