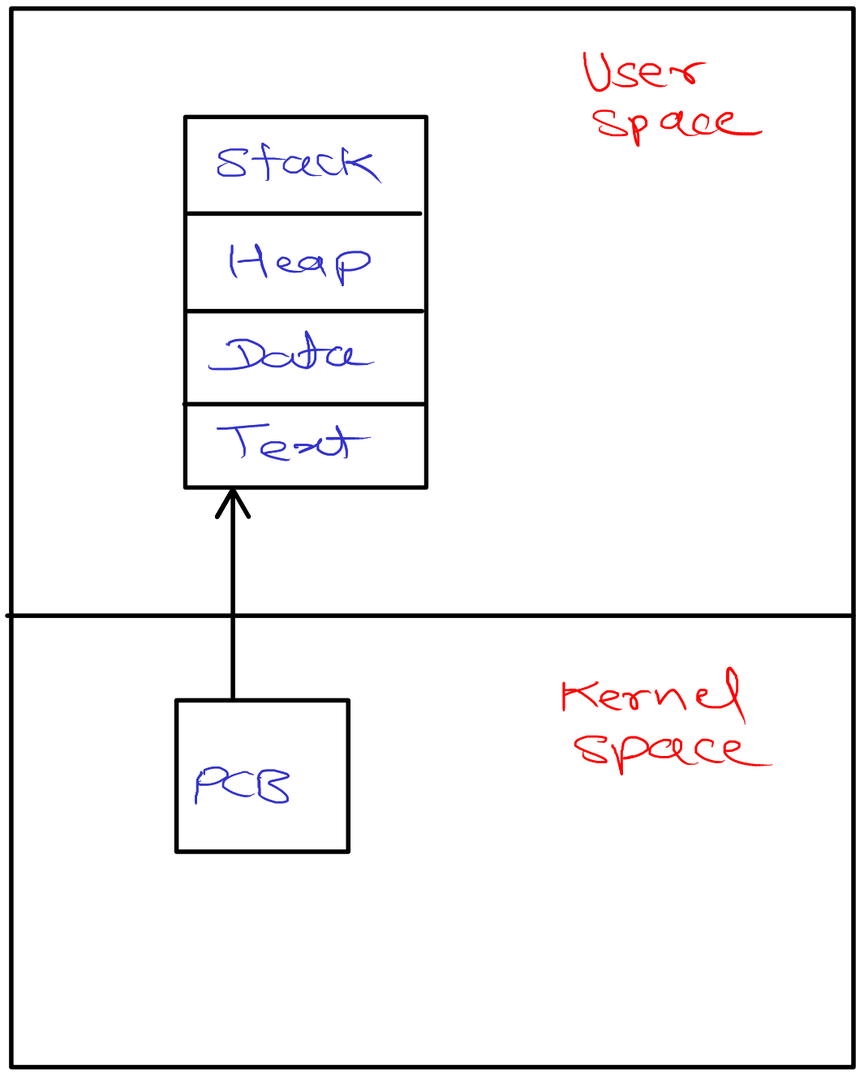
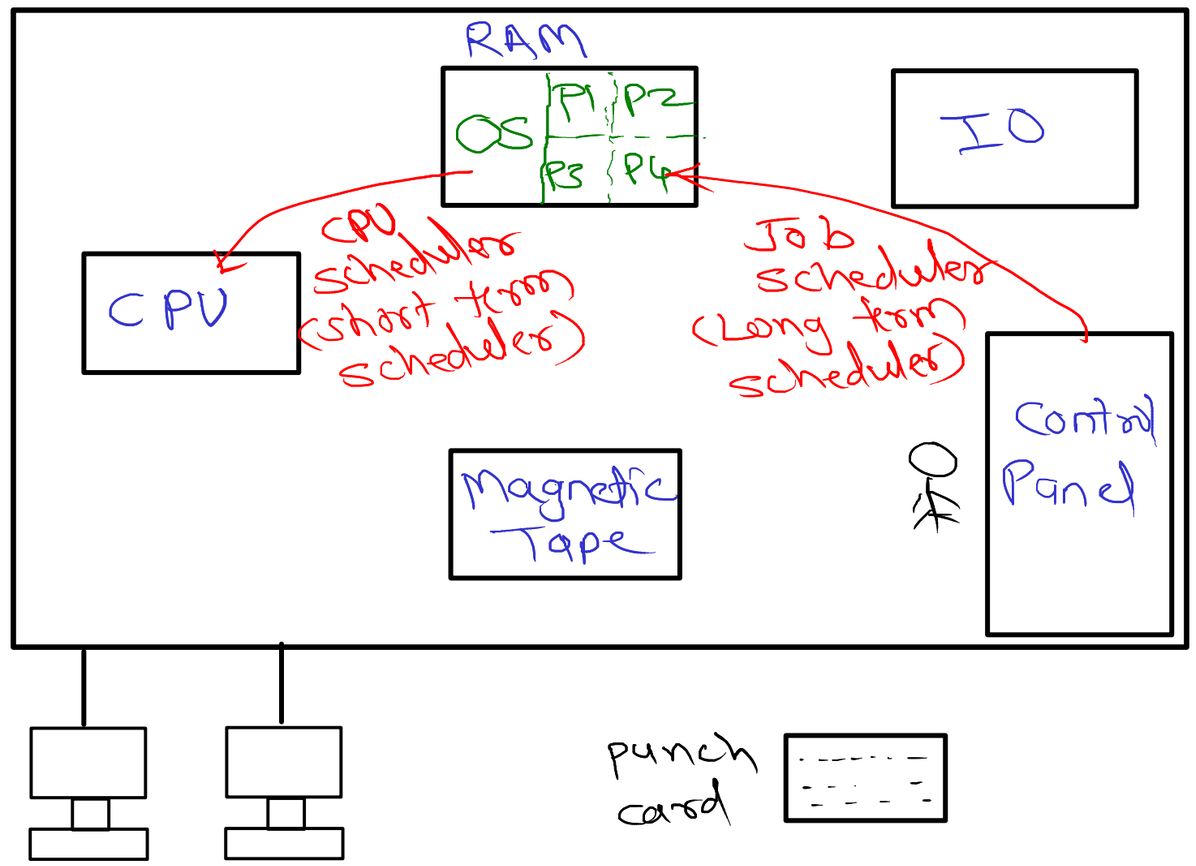
User space and Kernel Space



Types of Systems



1) Resident Monitor
2) Batch system
3) Multiprogramming system
   - Degree of Multiprogramming

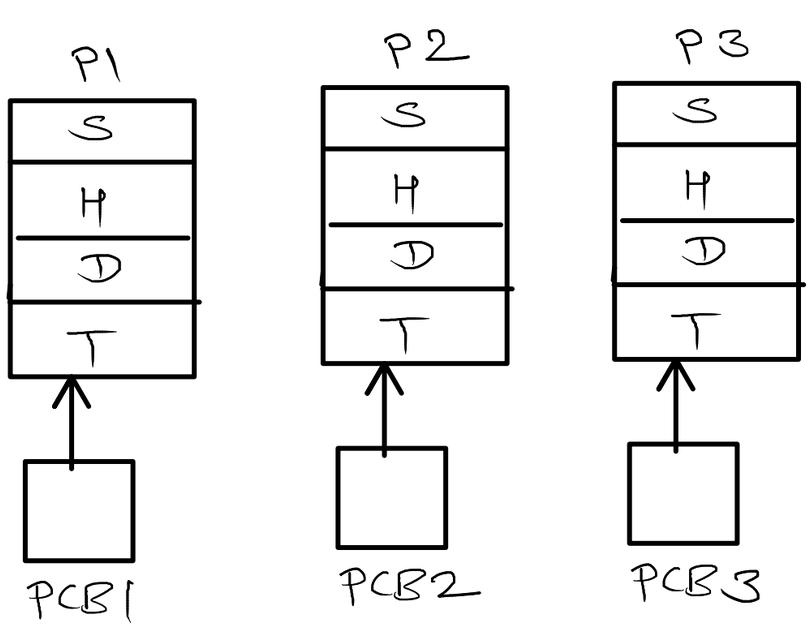
CPU time/burst - time spent on CPU
IO time/burst - time spent on IO
CPU burst > IO burst ⇒ CPU bound
IO burst > CPU burst ⇒ IO bound

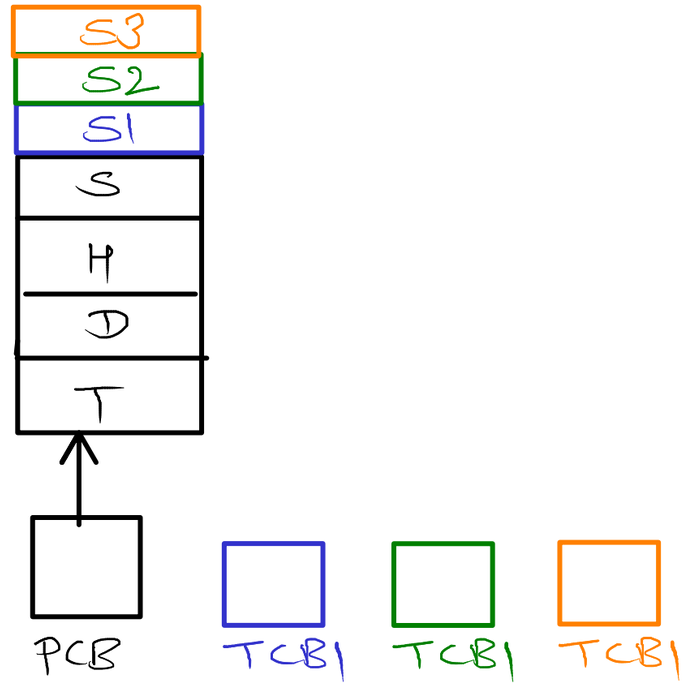
4) Time sharing system/ Multitasking system
Response time < 1 sec
1) Process based Multitasking
2) Thread based Multitasking (Multithreading)

5) Multiprocessing system

5) Multiuser system
multiple users can control the system

Process Vs Thread

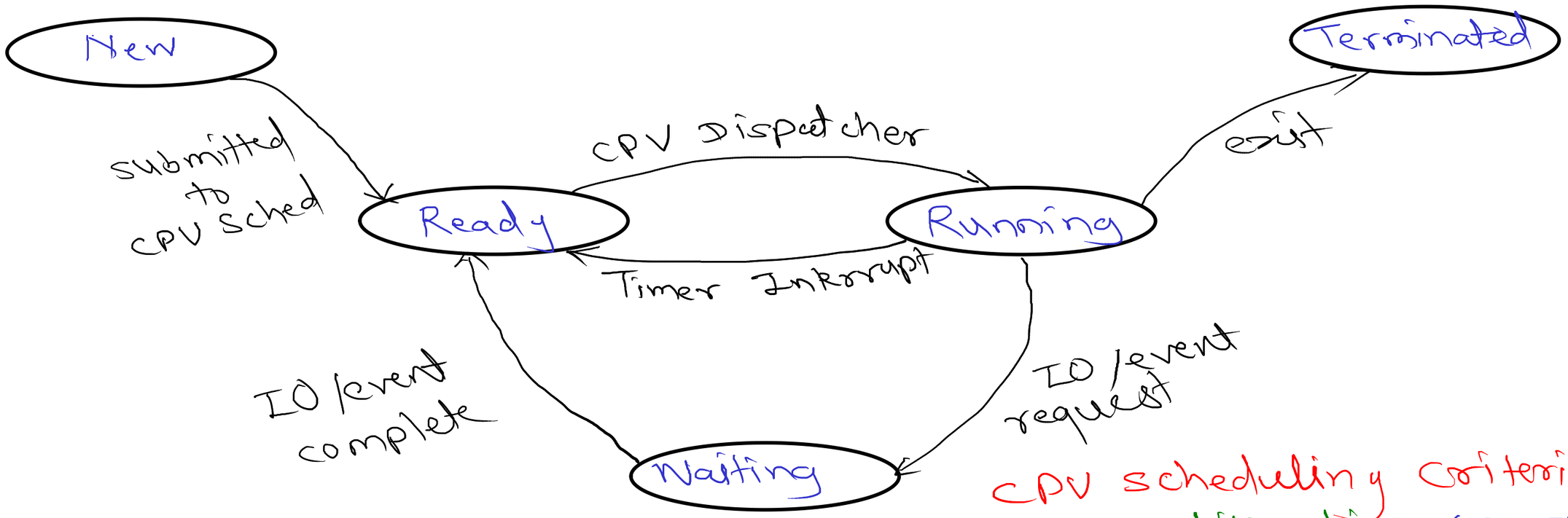




Thread – Lightweight process

- for every process one thread is created
- process is just a container of resources.
- the threads inside that container are scheduled on CPU

Process Life Cycle



Data structures of OS:

1) Ready queue:
   - all ready processes are kept

2) Job Queue/ Process table:
   - all processes of system.

3) Waiting queue:
   - processes which are waiting for IO/some event.
   - waiting queues are multiple (per device one)

CPU scheduling criterias:

1) CPU utilization: (max)

2) Throughput: (max)
   - amount of work done in unit time

3) Waiting time: (min)
   - total time spent by process into ready queue.

4) Response time: (min)
   - time from arrival into ready queue upto first time getting executed.

5) Turn Around time: (min)
   - total time spent into memory.

Types of Scheduling

1) Running → Terminated
2) Running → Waiting
3) Running → Ready
4) Waiting → Ready

} Non-Preemptive scheduling

} Preemptive scheduling

CPU Scheduling Algorithms

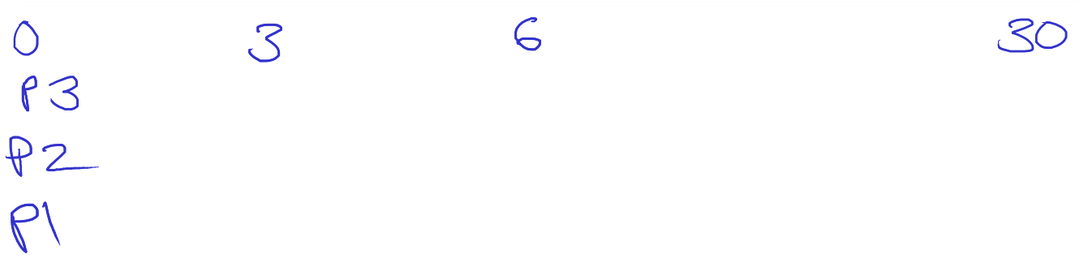
1) FCFS
2) SJF
3) Priority
4) RR
5) Fair Share

FCFS (First Come First Serve)

(Non-Preemptive scheduling)

Process	Arrival	CPU Burst	WT	RT	TAT
P1	0	24	0	0	24
P2	0	3	24	24	27
P3	0	3	27	27	30

Process	Arrival	CPU Burst	WT	RT	TAT
P3	0	3	0	0	3
P2	0	3	3	3	6
P1	0	24	6	6	30



Convoy's Effect

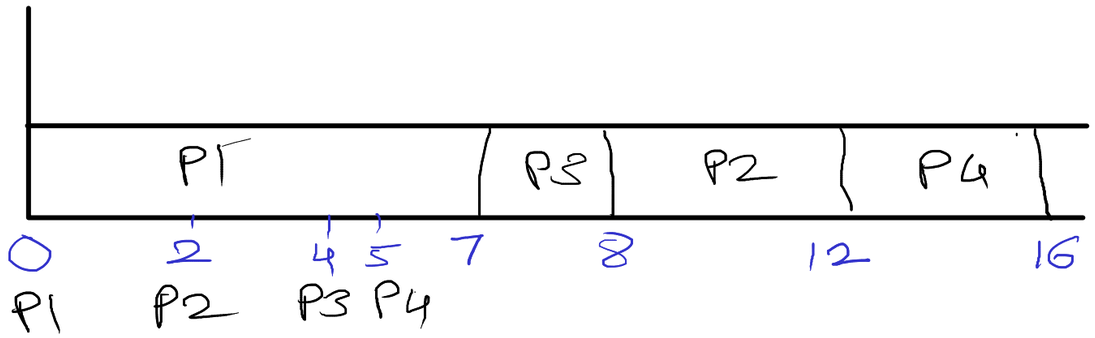
- due to arrival of longer process early into ready queue, all other processes has to wait for longer time

- we do not have any control on sequenece of processes

SJF (Shortest Job First)

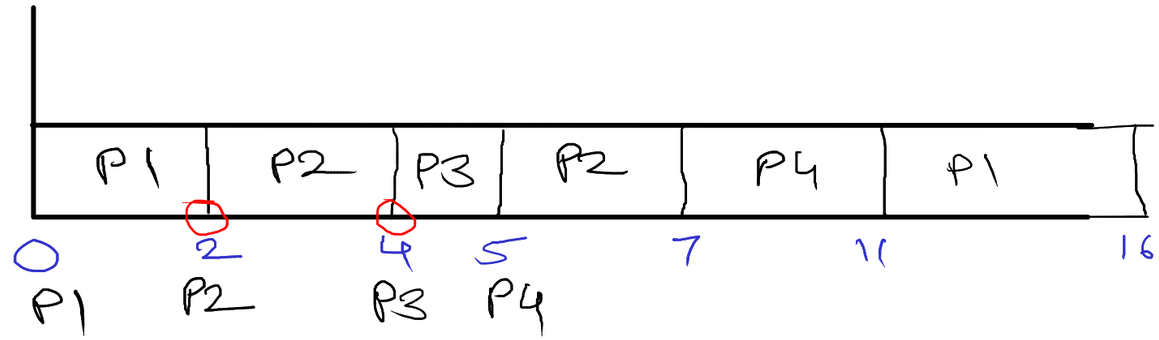
(Non Preemptive Scheduling)

Process	Arrival	CPU Burst	WT	RT	TAT
P1	0	7	0	0	7
P2	2	4	6	6	10
P3	4	1	3	3	4
P4	5	4	7	7	11



(Preemptive Scheduling)
(Shortest Remaining time First)

Process	Arrival	CPU Burst		WT	RT	TAT
P1	0	7	5	9	0	16
P2	2	4	2,0	1	0	5
P3	4	1	0	0	0	1
P4	5	4	4,0	2	2	6



Starvation

- due to longer CPU time process will get last chance to execute

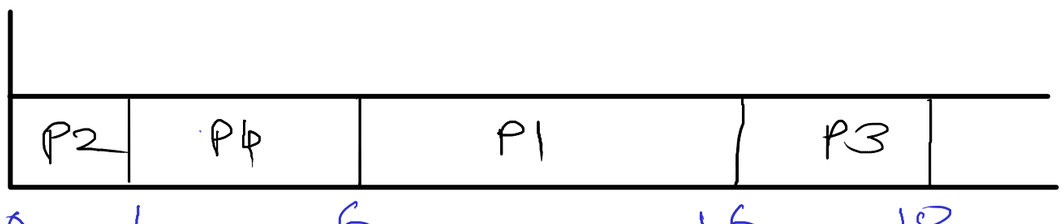
- there is no solution for starvation in SJF

Priority

(Non Preemptive Scheduling)

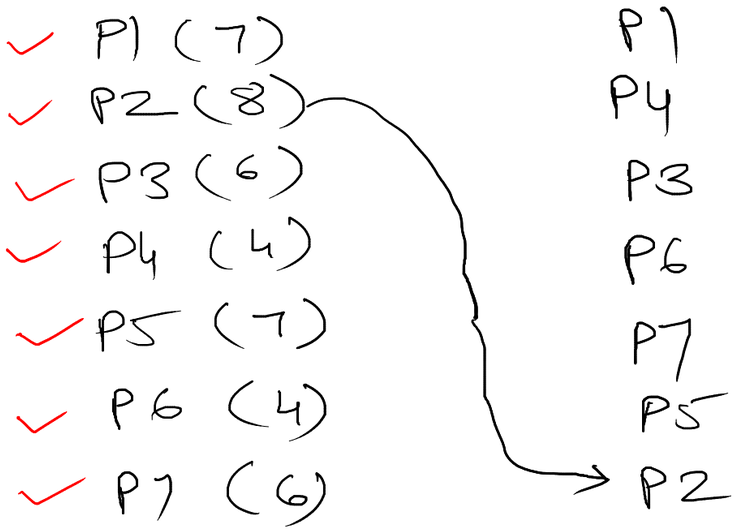
Process	Arrival	CPU Burst	Priority
P1	0	10	3
P2	0	1	1 (H)
P3	0	2	4 (L)
P4	0	5	2

	WT	RT	TAT
P1	6	6	16
P2	0	0	1
P3	16	16	18
P4	1	1	6



P1
P2
P3
P4

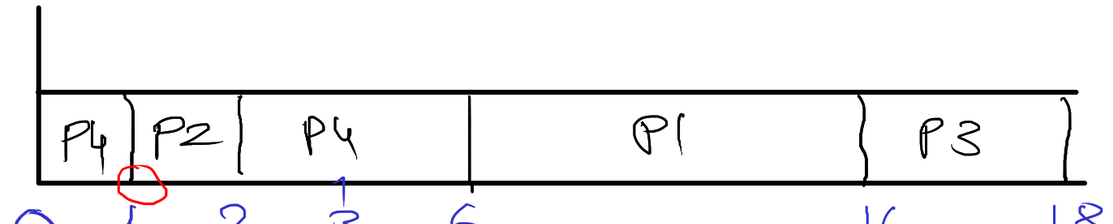
(Gantt's chart)



(Preemptive Scheduling)

Process	Arrival	CPU Burst	Priority
P1	0	10	3
P2	1	1	1
P3	3	2	4
P4	0	5	2

	WT	RT	TAT
10,6	6	6	
0	0	0	
2 10	13	13	
4 0	1	0	



P1 P2 P3
P4

Starvation

due to low priority, process will not get enough CPU time for execution

Aging

increase the priority of process gradually, so that it will get scheduled

RR (Round Robin)

(Pre emptive scheduling)

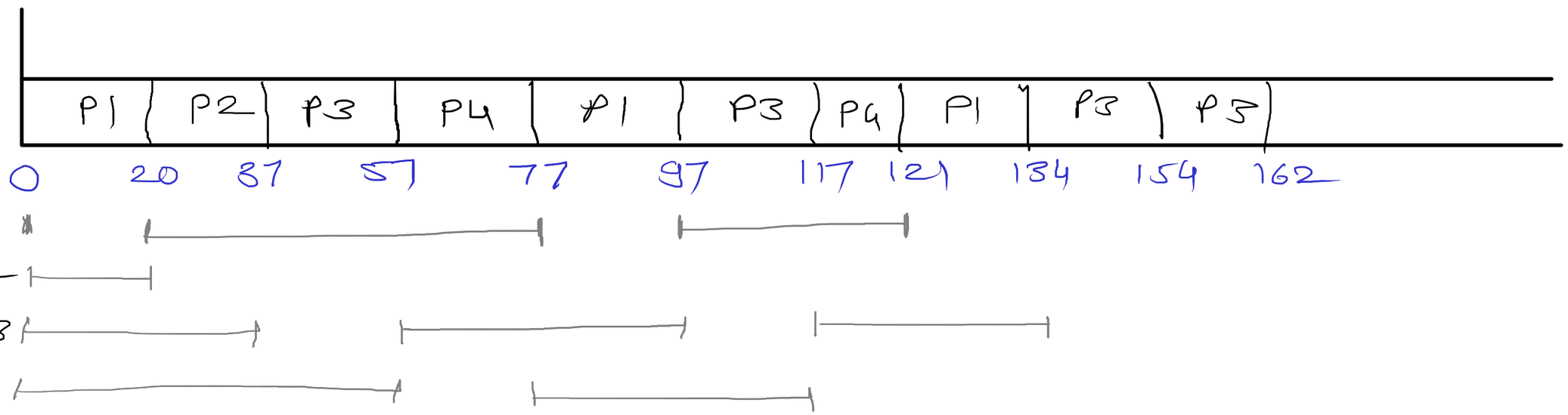
Process	CPU Burst
P1	53
P2	17
P3	68
P4	24

53, 13, 0
0
48, 28, 8
4, 0

WT	RT	TAT
0+57+24	0	134
20	20	37
37+40+17	37	162
57+40	57	121

Time Quantum
- CPU time is divided into equal parts.

TQ = 20



First waiting time is Response time

TQ = 100
↳ behave like FCFS

TQ = 4
↳ CPU overhead will increase.

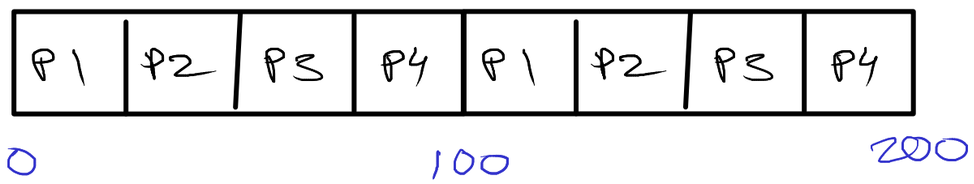
Fair Share

- CPU time is divided into time slices (epoch)
- some share of each epoch is given to the processes which are in ready queue.
- share is given to the process on the basis of their priority
- priority of every process is decided by its nice value
- nice values range ---> -20 to +19 (40 values)
  * -20 - highest priority
  * +19 - lowest priority

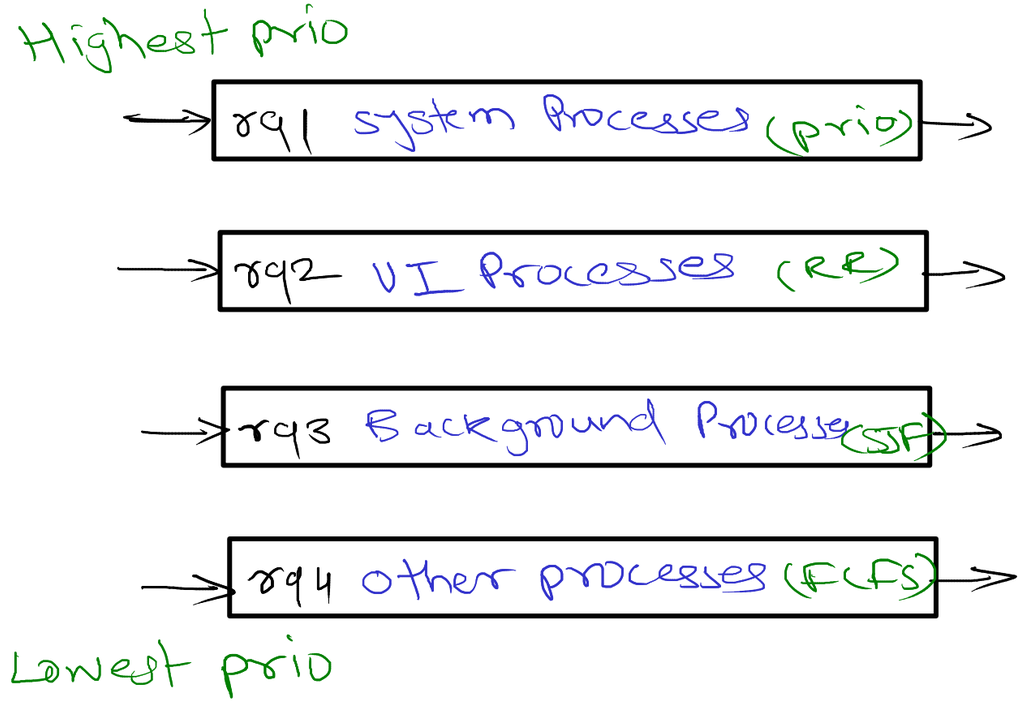
Process	Nice Value
P1	10
P2	10
P3	10
P4	10

Epoch - 100

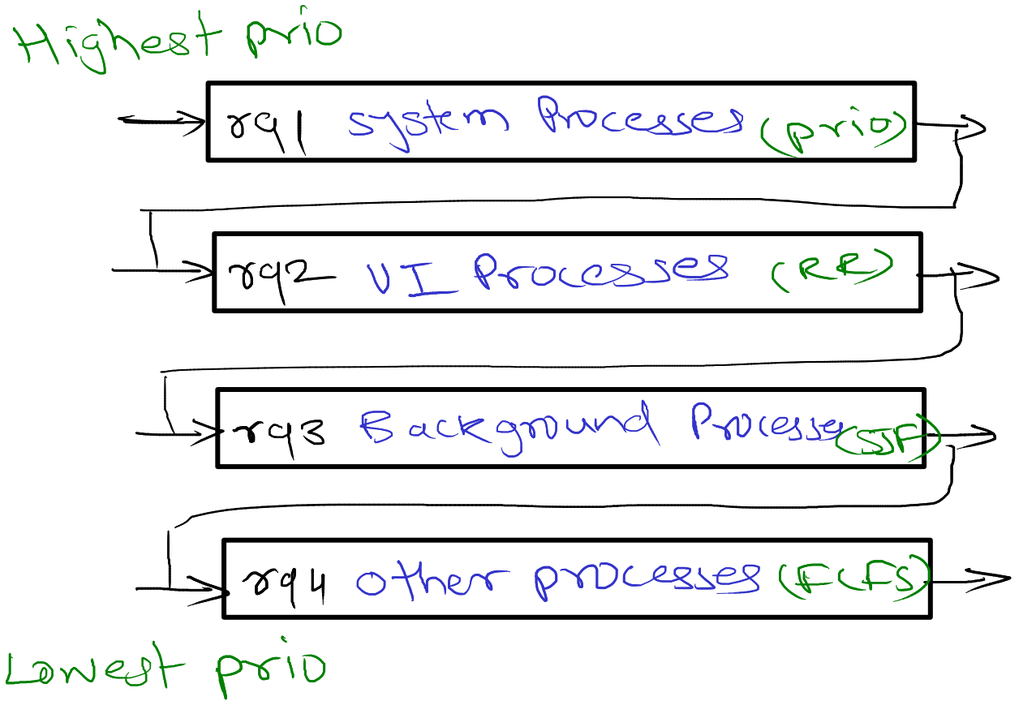
Process	Nice Value
P1	5
P2	5
P3	10
P4	10



Multi Level Ready Queue



Multi Level Feedback Ready Queue



Linux Scheduling

- There are two scheduling policies
    1. Non real time policy
        - i. SCHED_OTHER
        - ii. SCHED_BATCH
        - iii. SCHED_IDLE
    2. Real time policy
        - i. SCHED_FIFO
        - ii. SCHED_RR