#10

CPU - Scheduling
Chapter- 6

Nonpreemptive → can not be halted.
↳ [Atomic Execution]

Pre-emotive means can be forestalled

(*) CPU has alternating CPU and I/O burst cycles. A CPU has a short term scheduler / CPU scheduler, and its' job is to selects a process from memory and allocates the CPU to that process.
↓
(processes that are ready) i.e in Ready Queue

— Records ~~as~~ in ready queue are in .PCB (Process Control Block)
— Ready queue isn't necessarily a queue, it can be a priority queue, FIFO queue, tree.

Scheduling Criteria

→ CPU scheduling algorithm only affects waiting time.

(Min) → Turnaround, Response and Waiting time
Minimize

Convey Effect → Shorter process behind a longer process in FIFO queue.

Algorithms (Non-preemptive) — First Come Fist Serve (FCFS)
└ Shortest Job First (Ascending Sort) (SJF)

(All processes arrive at same time) Which is at time 0

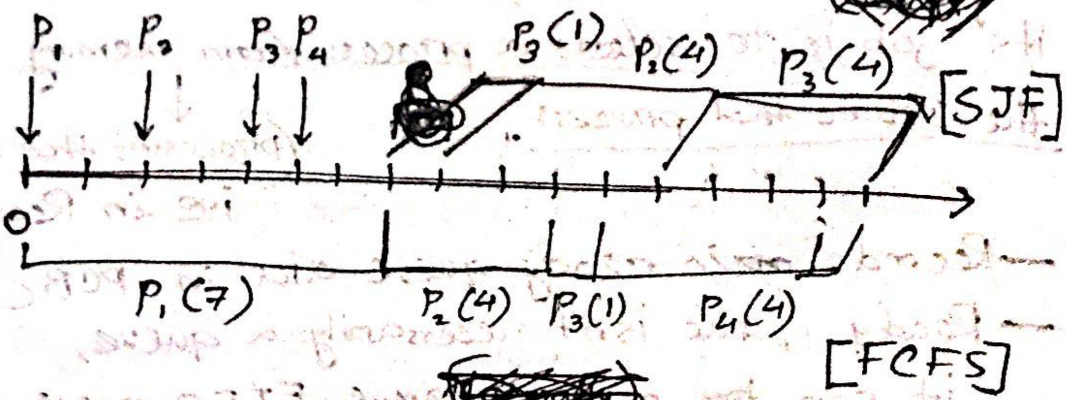
#11

Chapter-6

CPU-Scheduling

Non-Preemptive

	Arrival	Burst
P_1	0	7
P_2	2	4
P_3	4	1
P_4	5	4



In SJF after P_i completes, it looks for process with shortest burst which is P_3, then P_2 and then P_4.

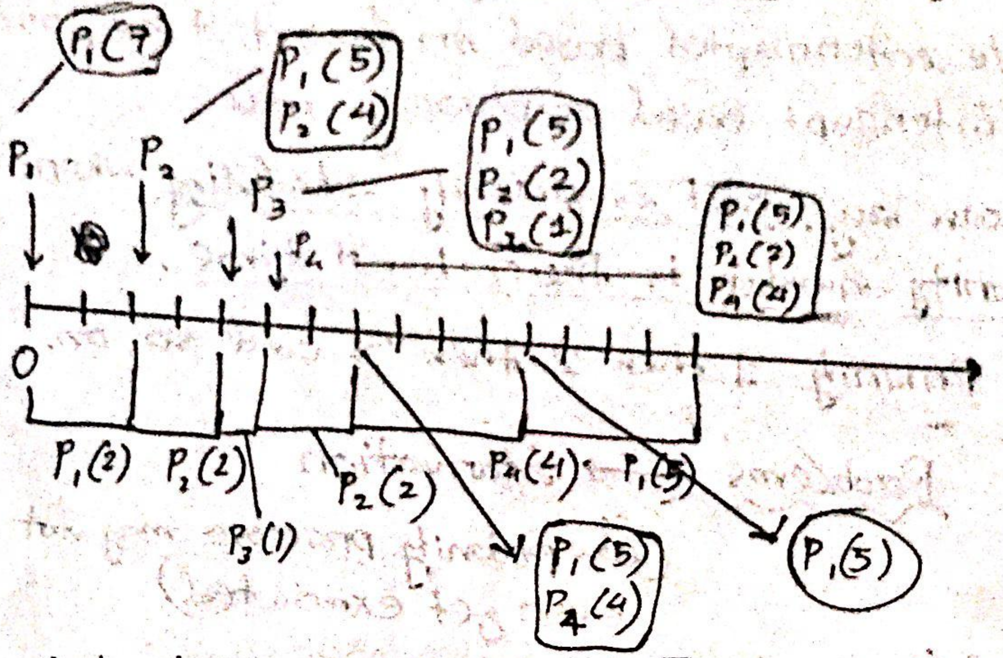
For SJF, waiting time,

$$P_1 = 0-0=0$$
$$P_2 = 8-2=6$$
$$P_3 = 7-4=3$$
$$P_4 = 12-5=7$$

(Serving time - Arrival time)

$$Avg = \frac{16}{4} = 4$$

Przeemptive (SJF)



Waiting time → [Response Times]

$$P_1 = (0-0) + (11-2) = 9$$

$$P_2 = (2-2) + (5-4) = 1$$

$$P_3 = (0-0) = 0$$

$$P_4 = (\cancel{7}-5) = \cancel{0}\ 2$$

Avg.

$$\frac{12}{4} = 3$$

$$W_t = (C_t - A_t) + \Sigma(CPU_A - CPU_{LR})$$

$$R_t = C_t - A_t$$

Response time → time till first response.

Pricority Scheduling

We internnupted based on burnst time, now we internnupt based on prionity value.

We can say, SJF is prionity scheduling where the prionity number is theirz burnst time.

→ Prionity 1 then 2 then 3 and so on.

Problems → Starvation
(low prionity processes may not even get executed)

Solution → Aging
(As time progresses, we increase the priority of a process i.e. decrease value of priority number).

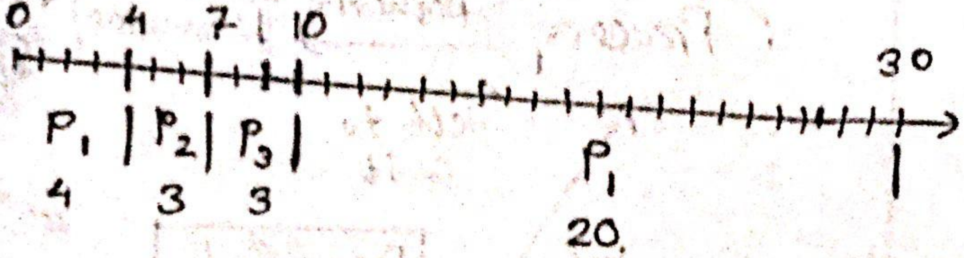
[Non-preemptive] Raund Robin (RR)

The communist system of serving process
Every process get a quanta time, and the processes are served in a circular queue.

Small time quanta → More context switching

Big time quanta → Higher response time.
(less like R.R more like normal nonpreemptive)

	B
P_1	~~24~~ 20
P_2	~~3~~ 0
P_3	~~3~~ 0



$q = 4$

Waiting time

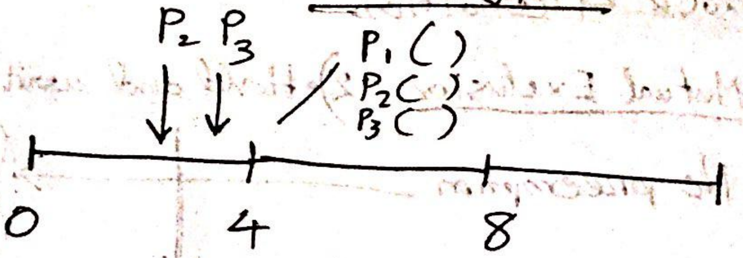
$P_1 = (0-0) + (10-4) = 6$

$P_2 = 4-0 = 4$

$P_3 = 7-0 = 7$

Higher turnaround time but [better response time.]

Mixture of round-robin and SJF



(R.R.)

- You can not interrupt in the middle of a quanta.
- But after a quanta, you sort and pick the smallest process.

Multi-level Queue

Each ~~process~~ queues get time slices

- Foreground Process → RR algorithm
- Background Process → FCFS algorithm.
  (First come first serve)