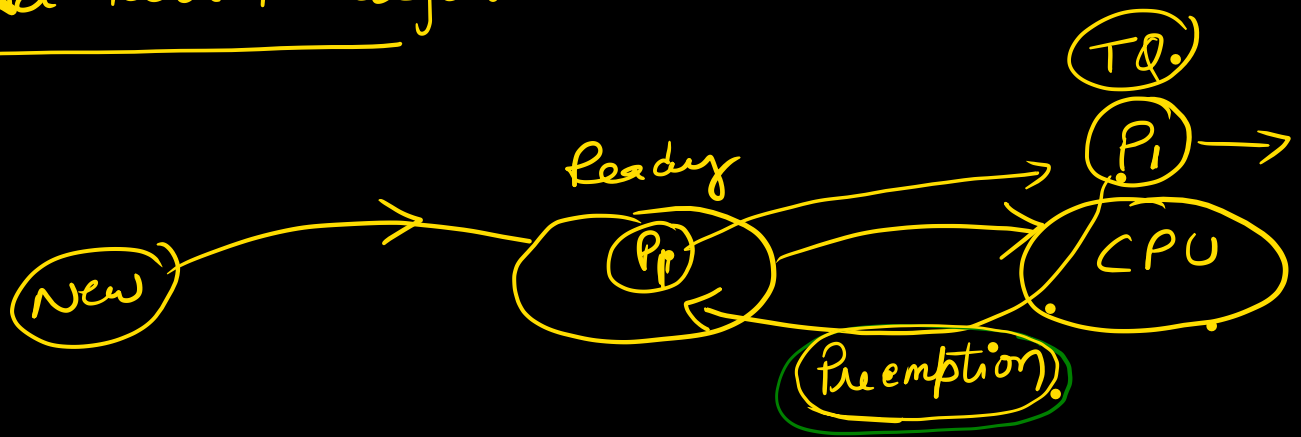
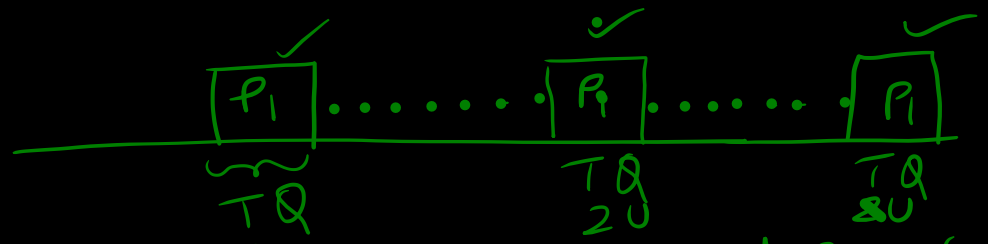
Round Robin algo:



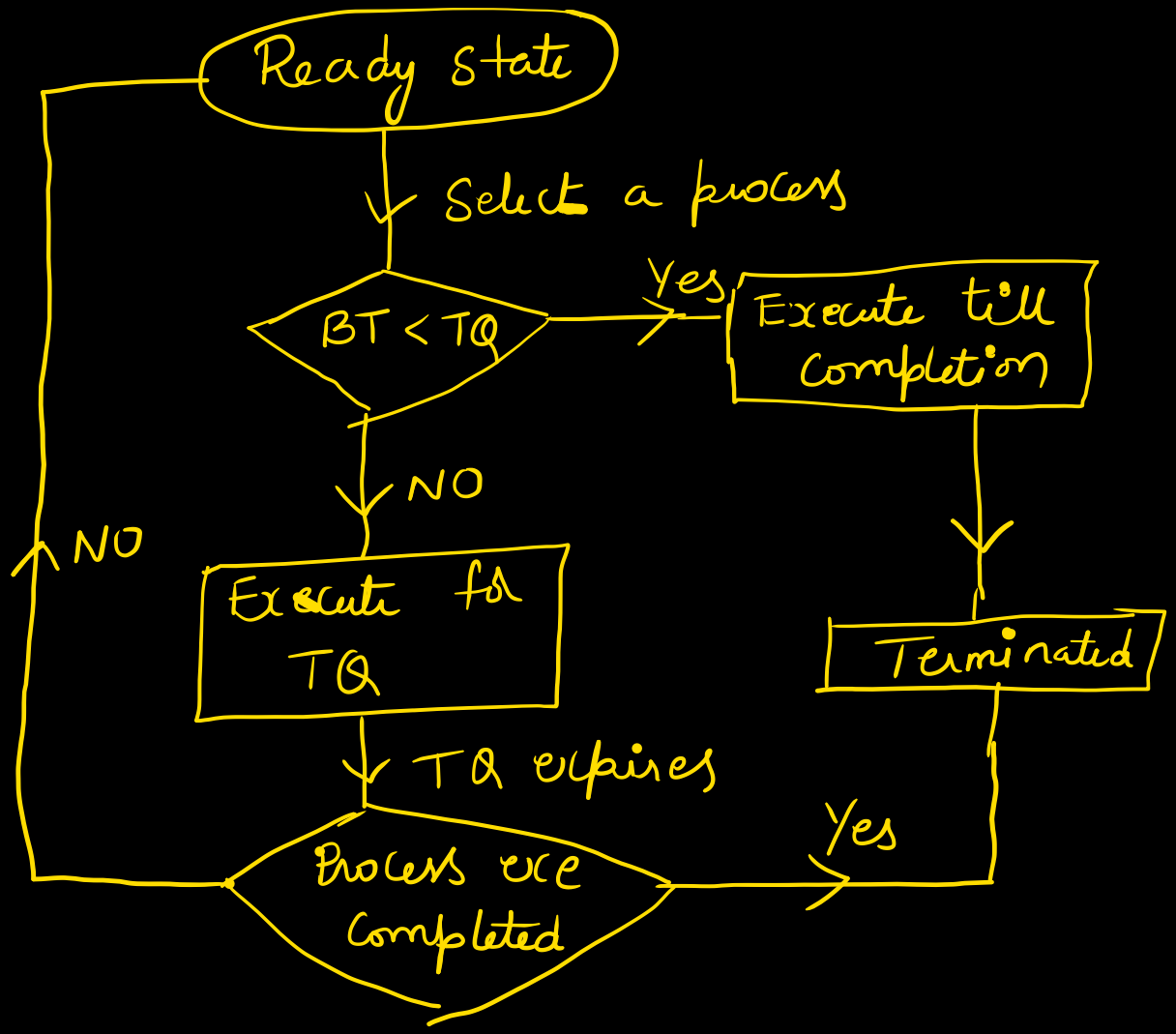
Run only for fixed time.
Called time quantum.

TQ = 1 U
= 2 U
3 U
4 U
⋮



If BT of P1 = 6 U
TQ = 2 U

flow chart of Round Robin:



RR:

Criteria: TQ + AT
↓
First come first served

mode: Preemptive.

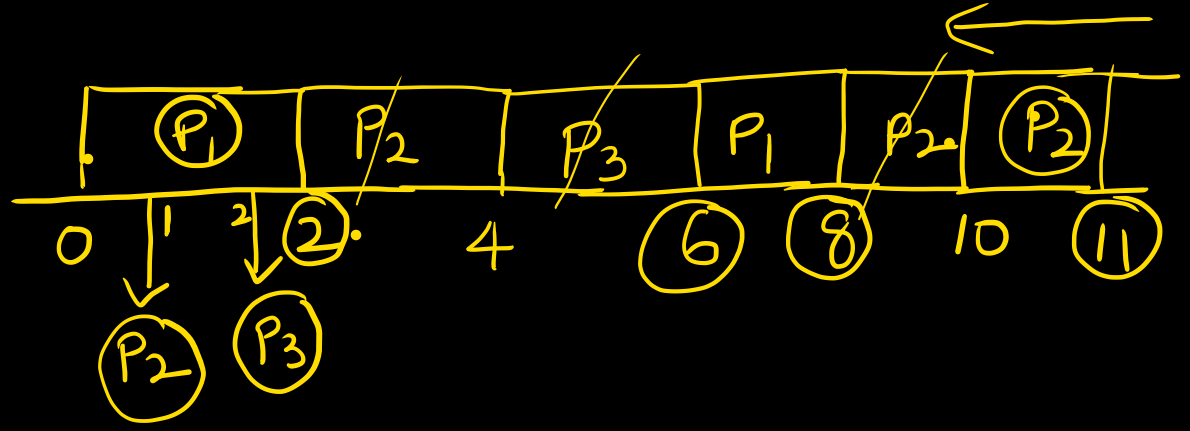
example.

Gate:

TQ = 2 units

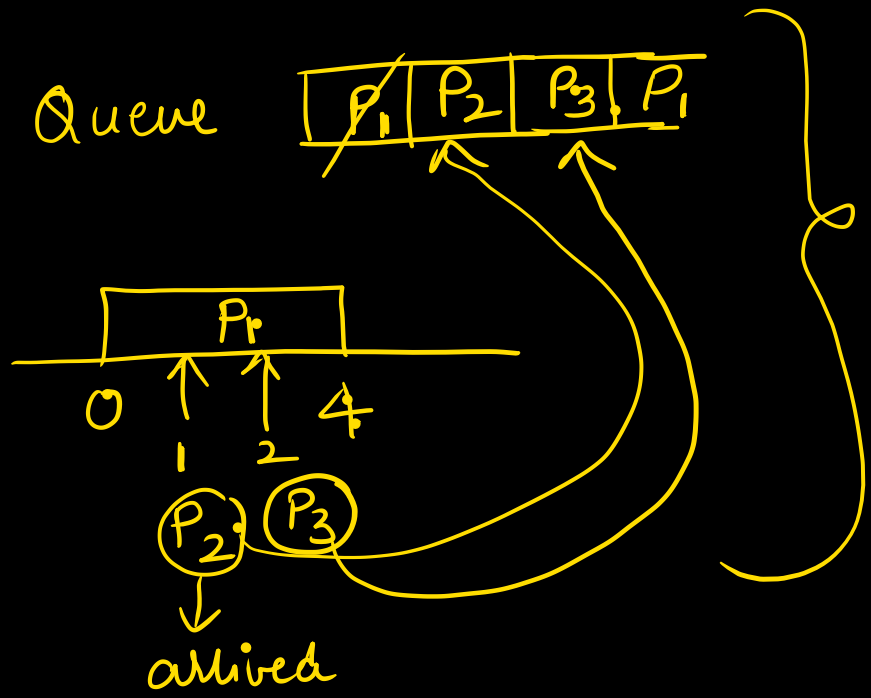
PNO	AT	BT	CT	TAT	WT
1	0	8	8	8	4
2	1	3	11	10	5
3	2	2	6	4	2

Queue: | P1 | P2 | P3 | P1 | P2 | P2 |



TQ = 4U

	AT	BT
1	0	~~4~~ 5 (1)
2	1	5
3	2	6.

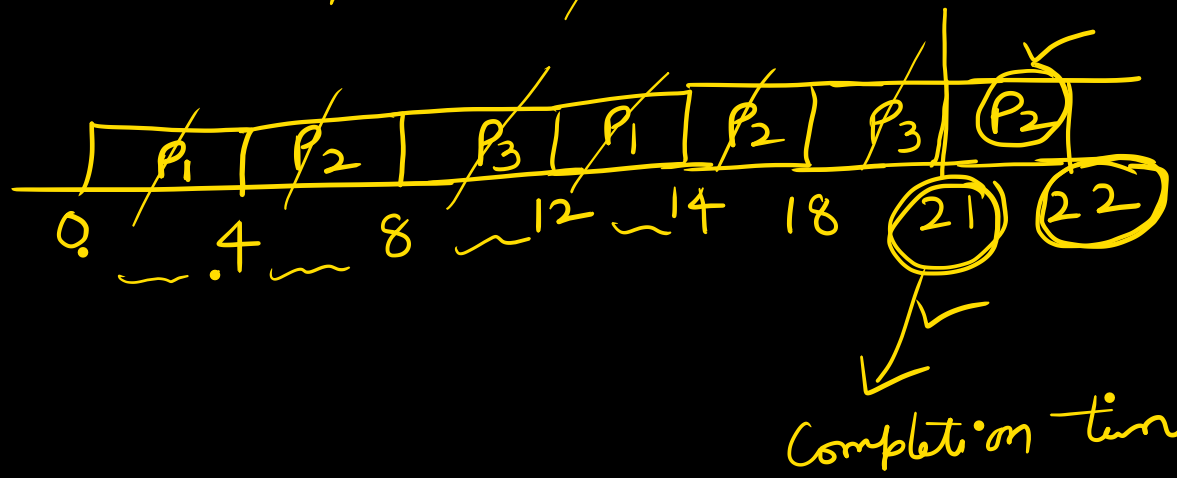


Gate:

PNO	AT	BT
1	0	6 ~~2~~ 0
2	1	~~4~~ ~~3~~ ~~1~~ 0
3	2	~~7~~ ~~3~~ 0

TQ = 4

Queue = | P1 | P2 | P3 | P1 | P2 | P3 | P2 |



Completion time of P3.

CT of P3

PNO	AT	BT
1	0	4 2 0
2	1	5 3 1 0
3	2	2 0
4	3	1 0
5	4	6 4 2 0
6	6	3 1 0

TQ = 2

Completion time of P_6

19

NO shortcut → You have to do procedure.

Queue | P_1 | P_2 | P_3 | P_1 | P_4 | P_5 | P_2 | P_6 | P_5 | P_2 | P_6 | P_5 |

TQ = 2 → 11 CS.

1 2 3 4 5 6 7 8 9 10 11

P_1	P_2	P_3	P_1	P_4	P_5	P_2	P_6	P_5	P_2	P_6	P_5

0 2 4 6 8 9 11 13 15 17 18 19 21

PNO	AT	BT
1	0	4 0
2	1	5 1 0
3	2	2 0
4	3	1 0
5	4	6 2 0
6	6	3 0

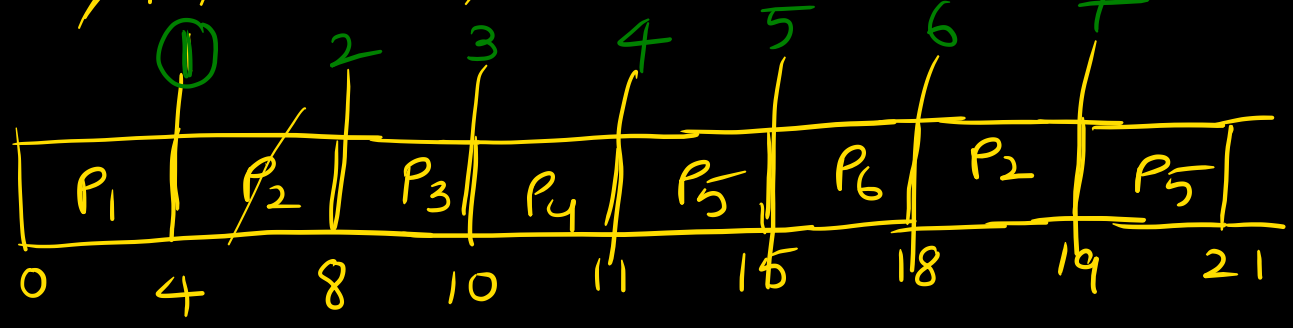
TQ = 4

CT of P_5 = ?

TQ → ∞

If TQ is ∞, then RR → FCFS ✓

Que P_1 P_2 P_3 P_4 P_5 P_6 P_2 P_5



TQ = 4 → 7 CS.

TQ = 2 → 11 CS.

∴ TQ ↑ → CS ↓ and Avg RT ↑

TQ ↓ → CS ↑ and Avg RT ↓

PNO	AT	BT
1	5	5
2	4	6
3	3	7
4	1	9
5	2	2
6	6	3

TQ = 3.

CT of P_3.

33

Break - 5 min.

RR is not based on Burst time. So it is used practically in many operating systems.

most popular.

TQ should not be too high because RT ↑

TQ should not be too low because context switching overhead will increase.

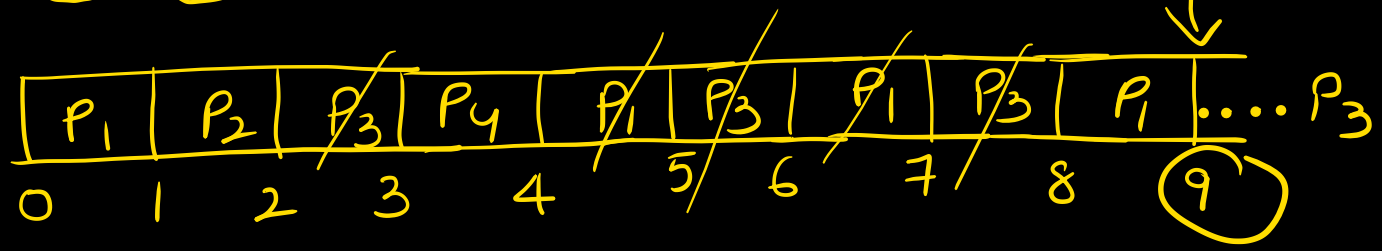
So experimentally TQ will be decided.

Gate Consider 4 Jobs P_1, P_2, P_3 and P_4 arriving in ready queue in the same order at time $t=0$. If the BT requirements of these jobs are 4, 1, 8, 1 respectively, what is CT of P1, assuming RR with TQ = 1.

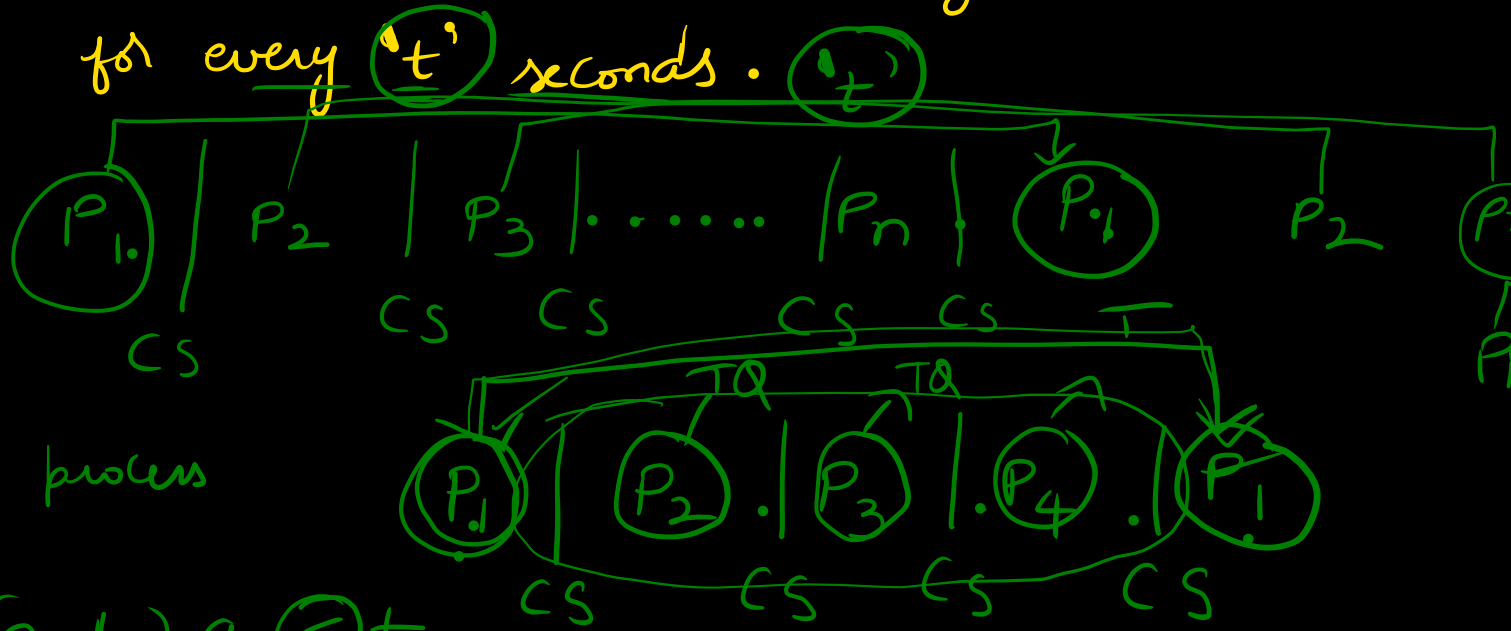
	AT	BT
P_1	0	4 3 2 1 0
P_2	0	1 0
P_3	0	8 7 6 5
P_4	0	1 0

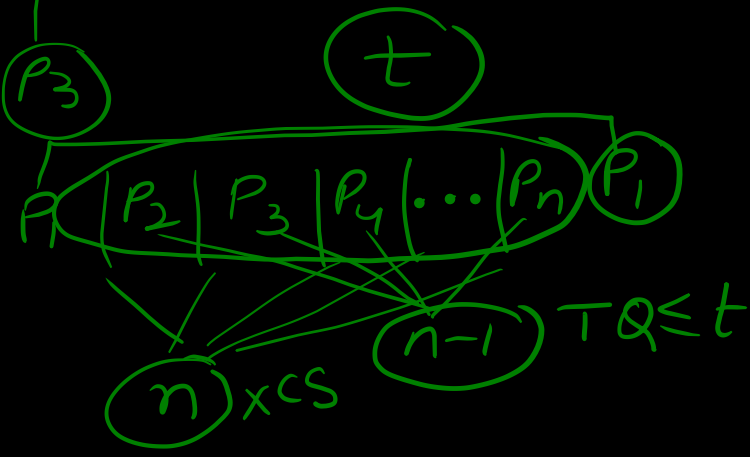
9

P1 Complete.



Gate: Consider 'n' processes sharing CPU in RR fashion. If the context switching time is 's' units, what must be the time quantum 'q' such that no of context switches are reduced, but at the same time each process is guaranteed to get the turn at CPU for every 't' seconds.





$nS + (n-1)q \leq t$

$q \leq \frac{t - ns}{(n-1)}$ ✓

$4CS + 3TQ$

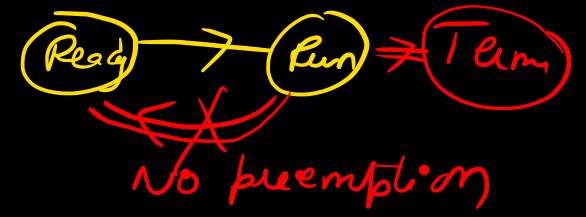
$4S + 3q \leq t$

Longest Job first (LTF)

Highest burst time.

Criteria = ~~Bust~~ Burst time

mode = non preemptive

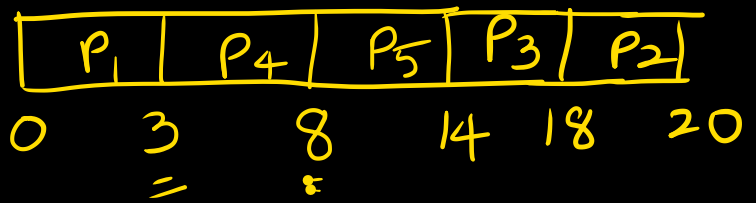


PNO	AT	BT
1	0	3
2	1	2
3	2	4
4	3	5
5	4	6

→ This is not practical to implement because of BT → We Cant know BT before executing.

→ It has convoy effect.

→

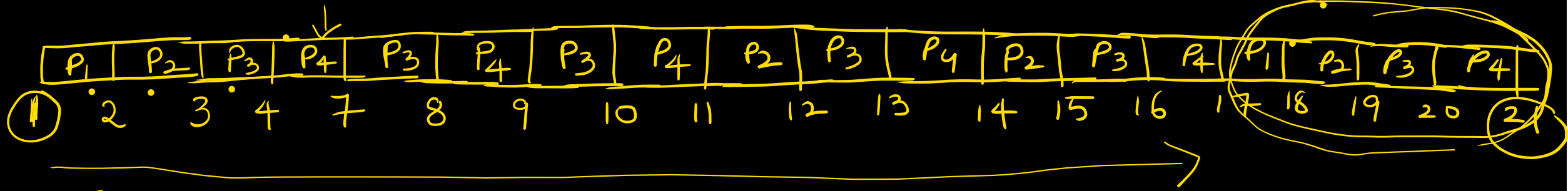


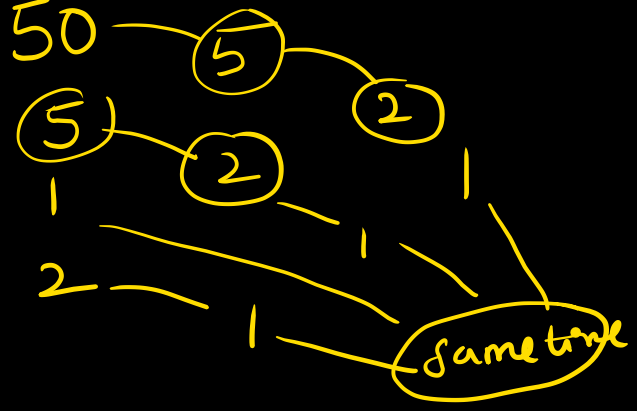
Longest remaining Time First:

criteria → BT + AT
Preemptive

PNO	AT	BT	CT	TAT	WT	RT
1	1	2 ~~1~~ 0	18	17	15	0
2	2	4 ~~3~~ ~~2~~ ~~1~~ 0	19	17	13	0
3	3	6 ~~5~~ ~~4~~ ~~3~~ ~~2~~ ~~1~~ 0	20	17	11	0
4	4	8 ~~7~~ ~~6~~ ~~5~~ ~~4~~ ~~3~~ ~~2~~ ~~1~~ 0	21	17	19	0

In LRTF all processes complete almost at same time.





why are we learning non practical algo?

Ans: Gate exams is asking these questions.

These algo are just used to compare each others performance.

Gate 06:

PNo	AT	BT	CT	TAT	WT	RT
1	0	2 0 X 0	12	12	10	8
2	0	4 3 2 X 0	13	13	9	4
3	0	8 4 3 2 X 0	14	14	6	0

What is avg TAT using LRTF?

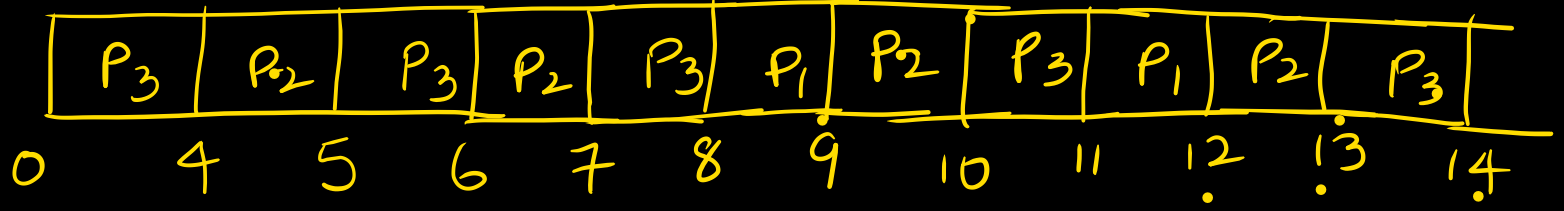
when BT are same go with AT

→ LRTF: CT are almost same.

AT → may be same ∧ diff

if almost same

→ TAT → almost same ✓



TAT → almost same.

if arrival time are almost same.

Completion times will be almost same