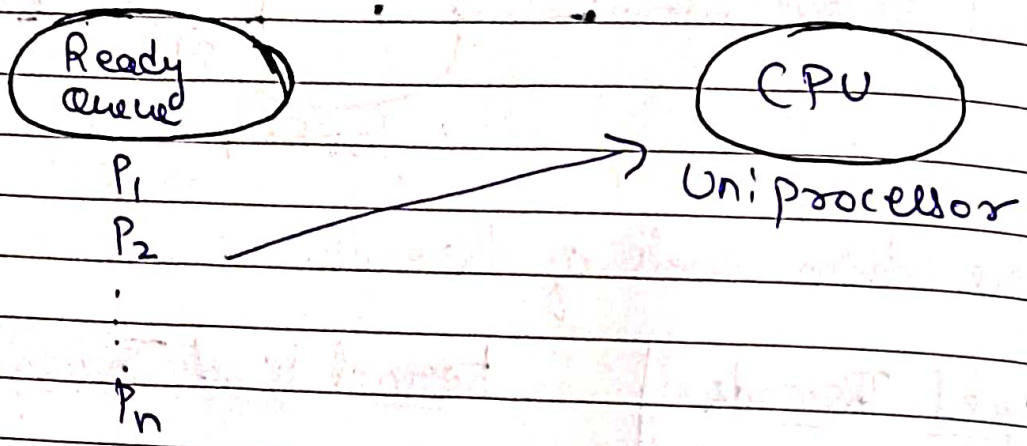
Process Scheduling Algorithms

Scheduling Algorithms is a way of selecting a process from ready queue and put it on a CPU.



Scheduling Algorithm

Pre-Emptive

- → SRTF (Shortest Job time remaining)
- → LRTF (Longest remaining time first)
- → Round Robin
- → Priority based

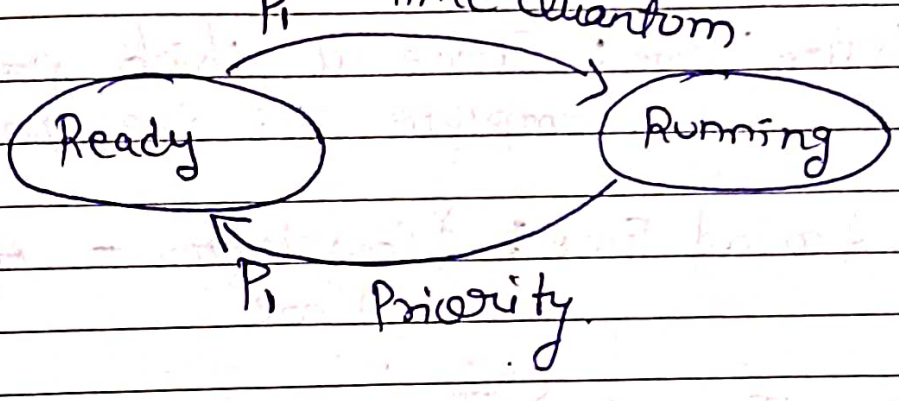
Non Pre-emptive

- → FCFS (First come first serve)
- → SJF (Shortest Job first)
- → LJF (Longest Job first)
- → HRRN (Highest Response Ratio next)
- → multi level queue

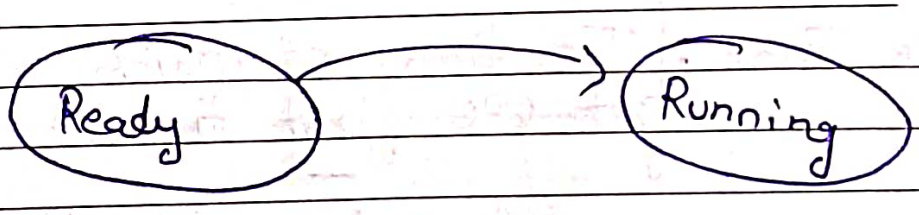
- **Preemptive**: Time-Quantum uses here. CPU can switch to another process in between the execution of the current process.

- **Non-Preemptive**: CPU completely executes the process then switch to another. No time-quantum.

Preemptive →



Non-preemptive →



NOTES

February '21

Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

CPU Scheduling

Arrival Time : The time at which process enter the Ready Queue or State.

Burst Time : Time required by a Process (Duration) to get execute on CPU.

Completion Time : Time at which process complete its execution.

Turn Around Time : { Completion Time − Arrival Time }

Waiting Time : { Turn Around Time − Burst time }

Response Time : [(The time at which a process get CPU first time) − (Arrival time)]

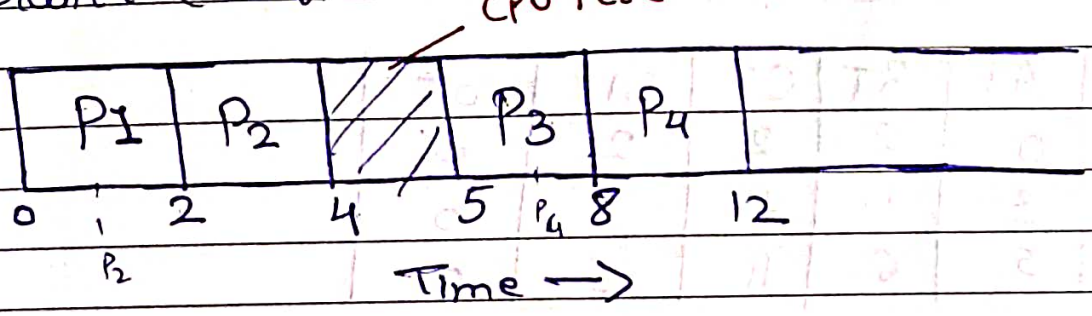
NOTES

First Come First Serve (FCFS) CPU Scheduling Algorithm

Criteria: "Arrival - Time"
Mode : "Non-Preemptive"

Process NO.	Arrival Time	Burst Time	Completion Time	TAT	WT	RT
P_1	0	2	2	2	0	0
P_2	1	2	4	3	1	1
P_3	5	3	8	3	0	0
P_4	6	4	12	6	2	2

Gantt Chart :-



NOTES

TAT = CT - AT
WT = TAT - BT
RT = CPU got - AT

RT = WT

P_1 = 0 - 0 P_3 = 5 - 5
P_2 = 2 - 1 P_4 = 8 - 6

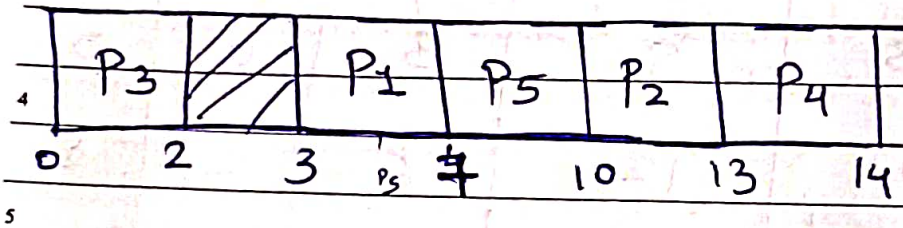
February '21

Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

• Example 2:-

Process	Arrival Time	Burst Time	TAT	WAT	CT
P_1	3	4	4	0	
P_2	5	3	8	5	7
P_3	0	2	2	0	13
P_4	5	1	4	8	2
P_5	4	3	6	3	14
					10

Gantt Chart:-

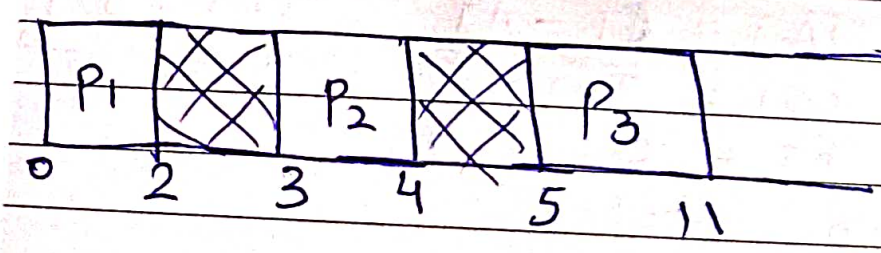


TAT = CT - AT

WT = TAT - BT

Ex 3:-

P	AT	BT	CT	TAT	WT
P_1	0	2	2	2	0
P_2	3	1	4	1	0
P_3	5	6	11	6	0



January '21

Su	Mo	Tu	We	Th	Fr	Sa
31					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

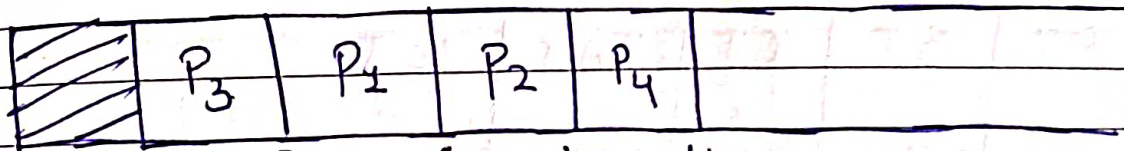
NOTES

Shortest Job First (SJF) Scheduling Algorithm -

Criteria : Burst time
Mode : Non-Preemptive

Process	Arrival Time	Burst Time	CT	TAT	WT	RT
P_1	1	3	6	5	2	2
P_2	2	4	10	8	4	4
P_3	1	2	3	2	0	0
P_4	4	4	14	10	6	6

Gantt Chart:-



$P_1 P_3$ $P_1 P_2$ $P_2 P_4$

NOTES In SJF, whose burst time is Shortest selected first
* If Burst time of two process is equal then check A.T. (Shortest prefered first)

February '21

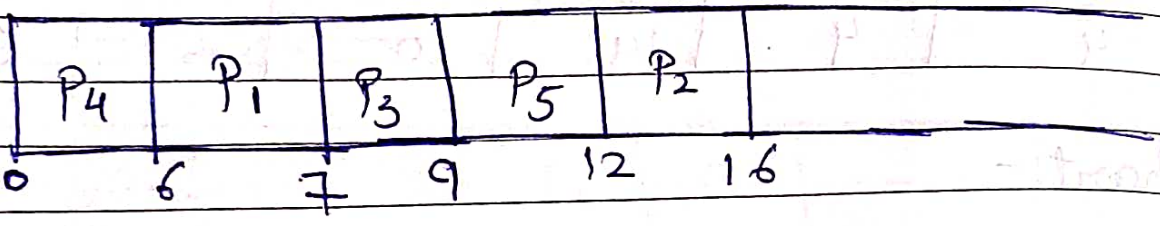
Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

29 FRIDAY

January
Week-05

• Ex 2:-

P	AT	BT	CT	TAT	WT
P_1	3	1	7	4	3
P_2	1	4	16	15	11
P_3	4	2	9	5	3
P_4	0	6	6	6	0
P_5	2	3	12	10	7



Shortest Remaining Time First (SJF with Preemption)

Criteria: Burst Time
mode: Preemptive

Process	Arrival Time	Burst Time	TAT	WT	RT	CT
P_1	0	~~5~~ 3 ~~2~~ 1	9	4	0	9
~~P_2~~	1	3 ~~2~~ 1	3	0	0	4
P_3	2	4	11	7	7	13
~~P_4~~	4	~~1~~	1	0	0	5

Grantt Chart:-

P_1	P_2	P_2	P_2	P_4	P_1	P_3	P_1	P_1	P_3
0	1	2	3	4	5	6	7	8	9	13

SUNDAY 31

P → P_1P_2 $P_1P_2P_3$ $P_1P_2P_3$ $P_1P_3P_4$ P_1P_3 P_1P_3
BT → 4 3 4 2 4 4 1 4 4 4 1 4 4 3 4

In Preemptive RT ≠ WT

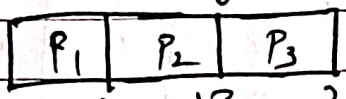
NOTES

February '21

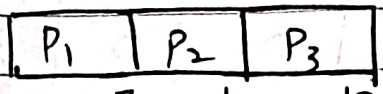
Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

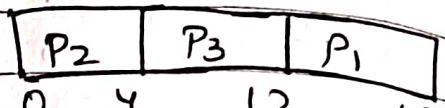
Process	Arrival time	Burst time
P_1	0	7 6 5
P_2	1	4 3 2 1
P_3	2	8

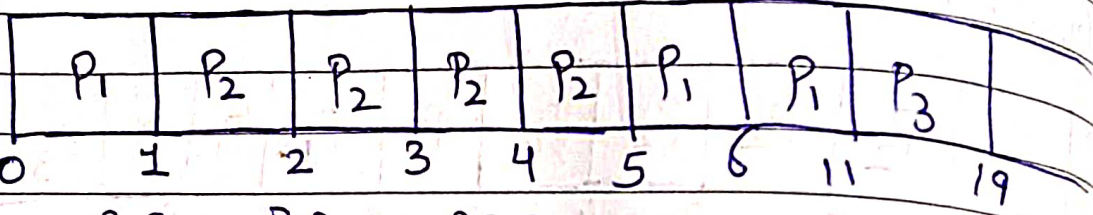
The Gantt Chart for Preemptive SJF Scheduling algorithm is — ?

1) 

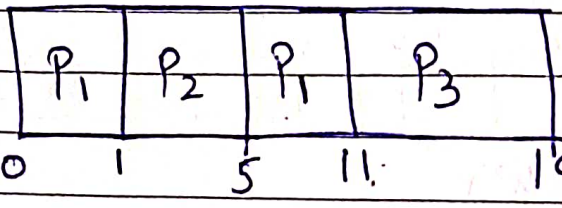
(2) 

3) 

(4) 

Sol) 

P_1P_2 $P_1P_2P_3$ $P_1P_2P_3$



NOTES

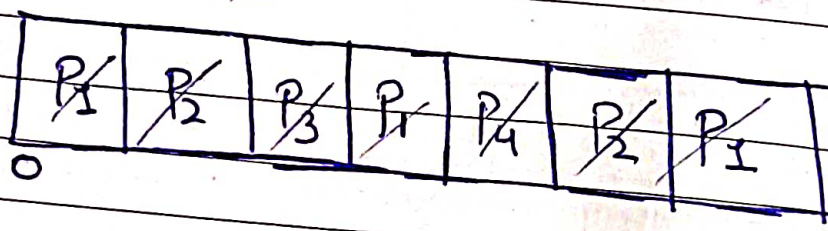
• Round Robin CPU Scheduling –

Criteria: Time Quantum
Mode: Preemptive

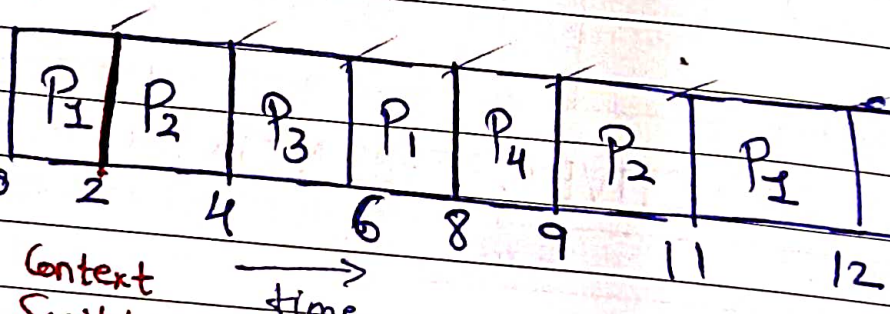
Process	Arrival Time	Burst Time	CT	TAT	WAT	RT
P1	0	5 ✗ 10	12	12	7	0
P2	1	4 ✗ 0	11	10	6	1
P3	2	2 ✗ 0	6	4	2	2
P4	4	1 ✗ 0	9	5	4	4

Given T.Q. = 2

Ready Queue



Running Queue



Context Switching
→ Save Running Process and send to ready queue and then load new process.

How many context Switches: 6

Pre-emptive Priority Scheduling Algorithm

Criteria: Priority
Mode : Preemptive

Process	Process No.	AT	BT	CT	TAT	WT
10	P_1	0	5 4	12	12	7
20	P_2	1	4 3 0	8	7	3
30	P_3	2	2 1 0	4	2	0
40	P_4	4	1 0	5	1	0

Higher the no. Higher the priority

P_1	P_2	P_3	P_3	P_4	P_2	P_1	

0 1 2 3 4 5 8 12

P_1P_2 $P_1P_2P_3$ $P_1P_2P_3$ $P_1P_2P_4$ P_1P_2

Ex:-

P.id	At	Bt	Prio.	CT	TAT	WT
P_1	0	4 3 0	2	15	15	11
P_2	1	3 2 0	3	12	11	8
P_3	2	1 0	4	13	1	0
P_4	3	5 4 3 0	5	8	5	0
P_5	4	2 0	5	10	6	4

NOTES

P_1	P_2	P_3	P_4	P_4	P_4	P_5	P_2	P_1

0 1 2 3 4 5 8 10 12 15

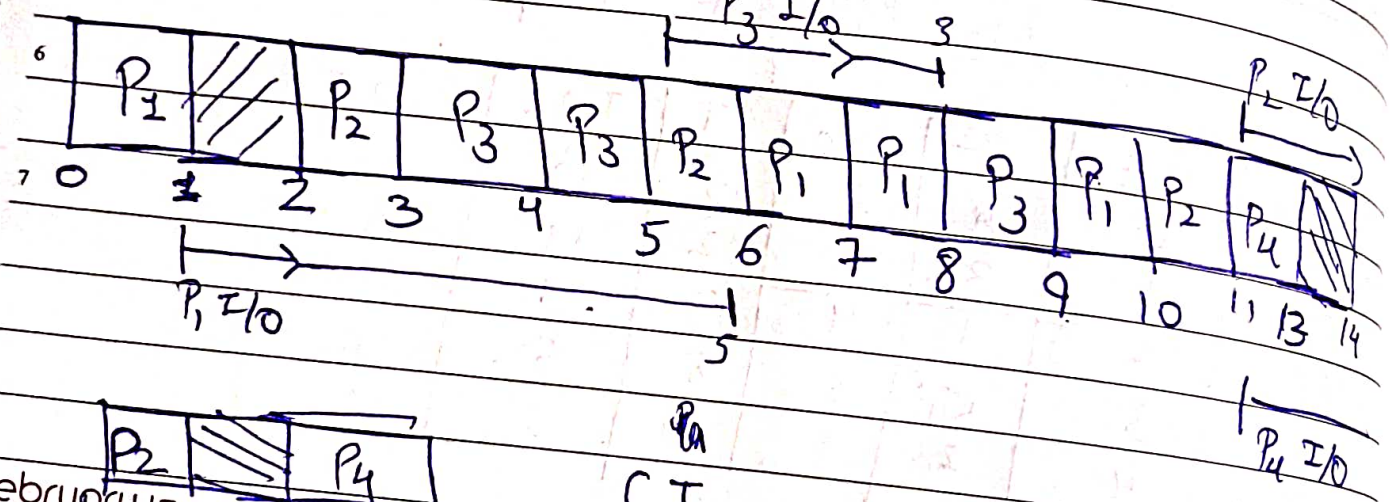
P_1P_2 $P_1P_2P_3$ $P_1P_2P_4$ $P_1P_2P_4P_5$ $P_1P_2P_1$

• Mix Burst time (CPU & I/O both) in CPU Scheduling

Process	Arrival Time	Priority	CPU	I/O	CPU
P1	0	2	10	5	3 2 6
P2	2	3	3 2 1	3 0	10
P3	3	1	2 1 0	3 0	10
P4	3	4	20	4	1

Mode: Preemptive
Criteria: Priority based [low PD. high Prior]
Find CT of P1, P2, P3, P4

Gantt chart:



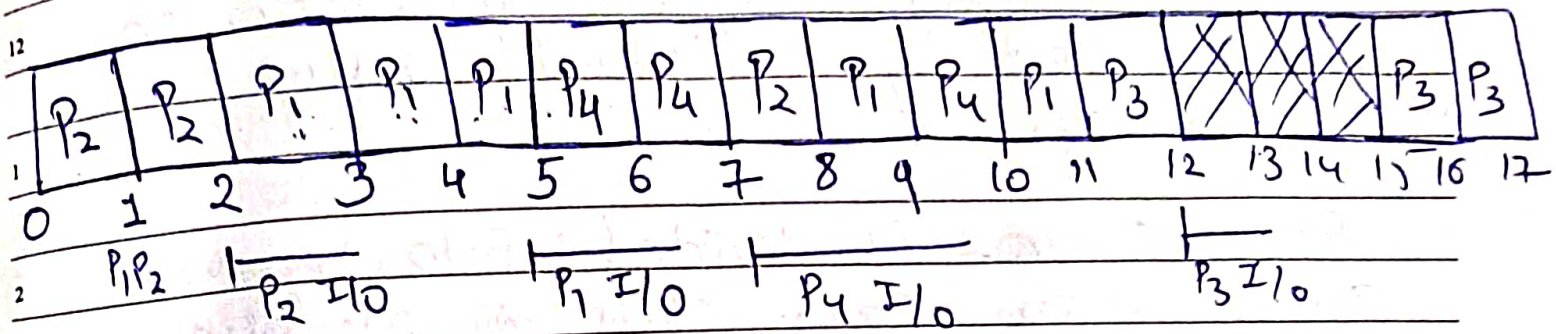
	CT
P1	10
P2	15
P3	9
P4	18

CPU idleness: 4/18

usage

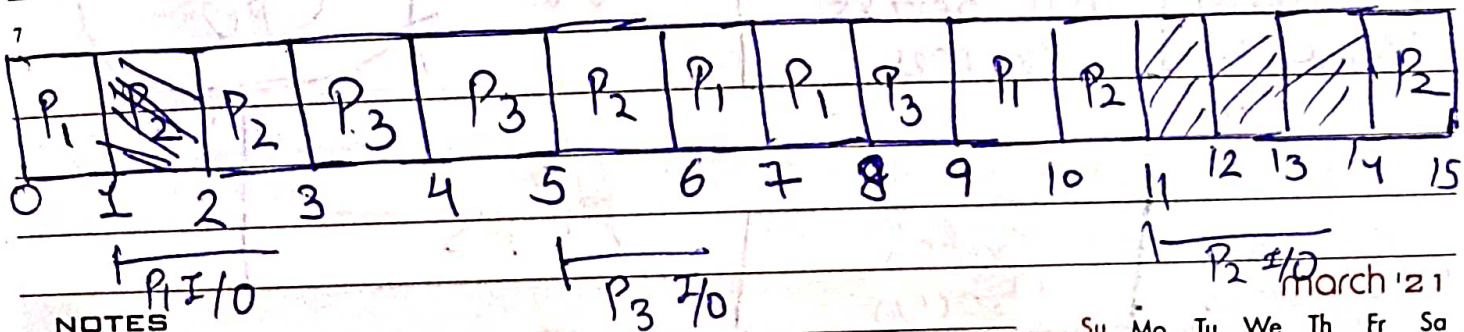
Ex2:-

P.no	A.T	CPU	I/O	CPU	Priority	CT
P1	0	3 2 1	2 0	2 1 0	2	11
P2	0	2 1 0	4 0	1 0	3	8
P3	2	1 0	3 0	2 0	1	17
P4	5	2 1 0	2 0	1 0	4	10



Ex3:-

P.no.	AT	Prior.	CPU	I/O	CPU	CT
P1	0	2	1 0	5 0	3 2 1 0	10
P2	2	3	3 2 1	3 0	1 0	15
P3	3	1	2 1 0	3 0	1 0	9

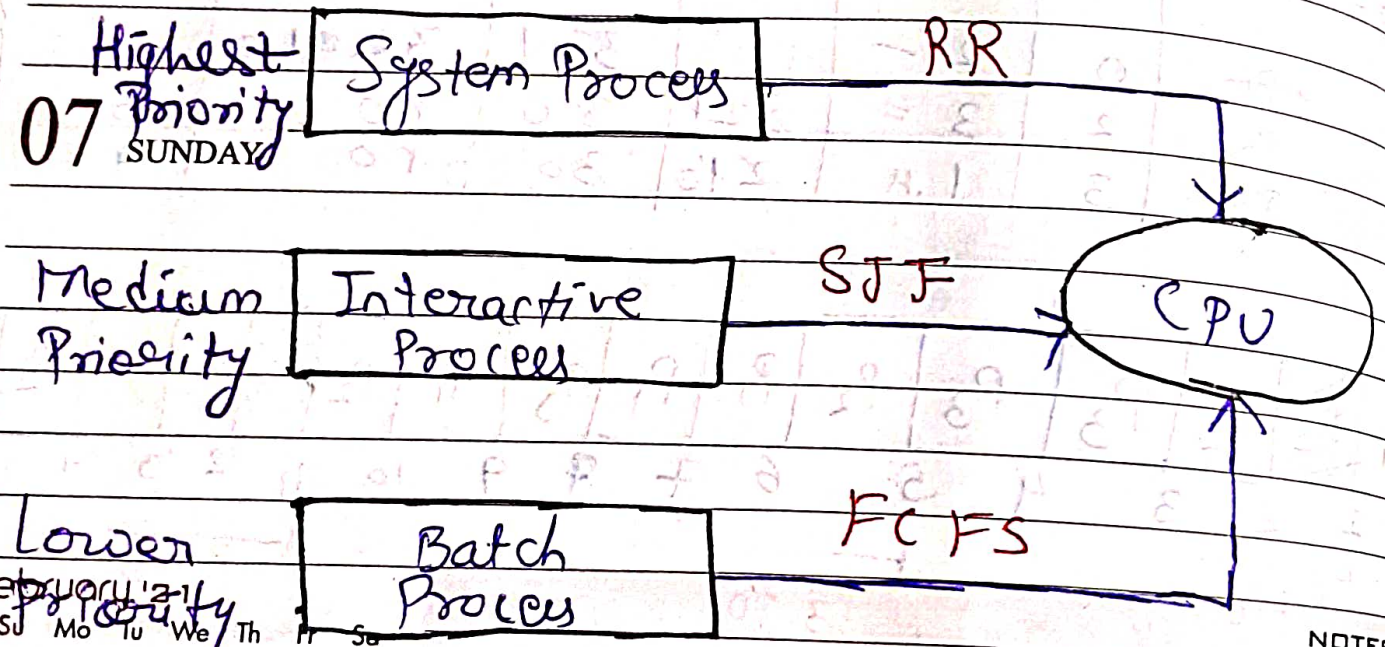


NOTES

Multilevel Queue Scheduling

It may happen that processes in the ready queue can be divided into different classes where each class has its own scheduling needs. These classes This kind of situation is handled by multilevel queue scheduling.

Ready Queue is divided into separate queues for each class of processes. And each queue has its own scheduling algorithm.



07 SUNDAY

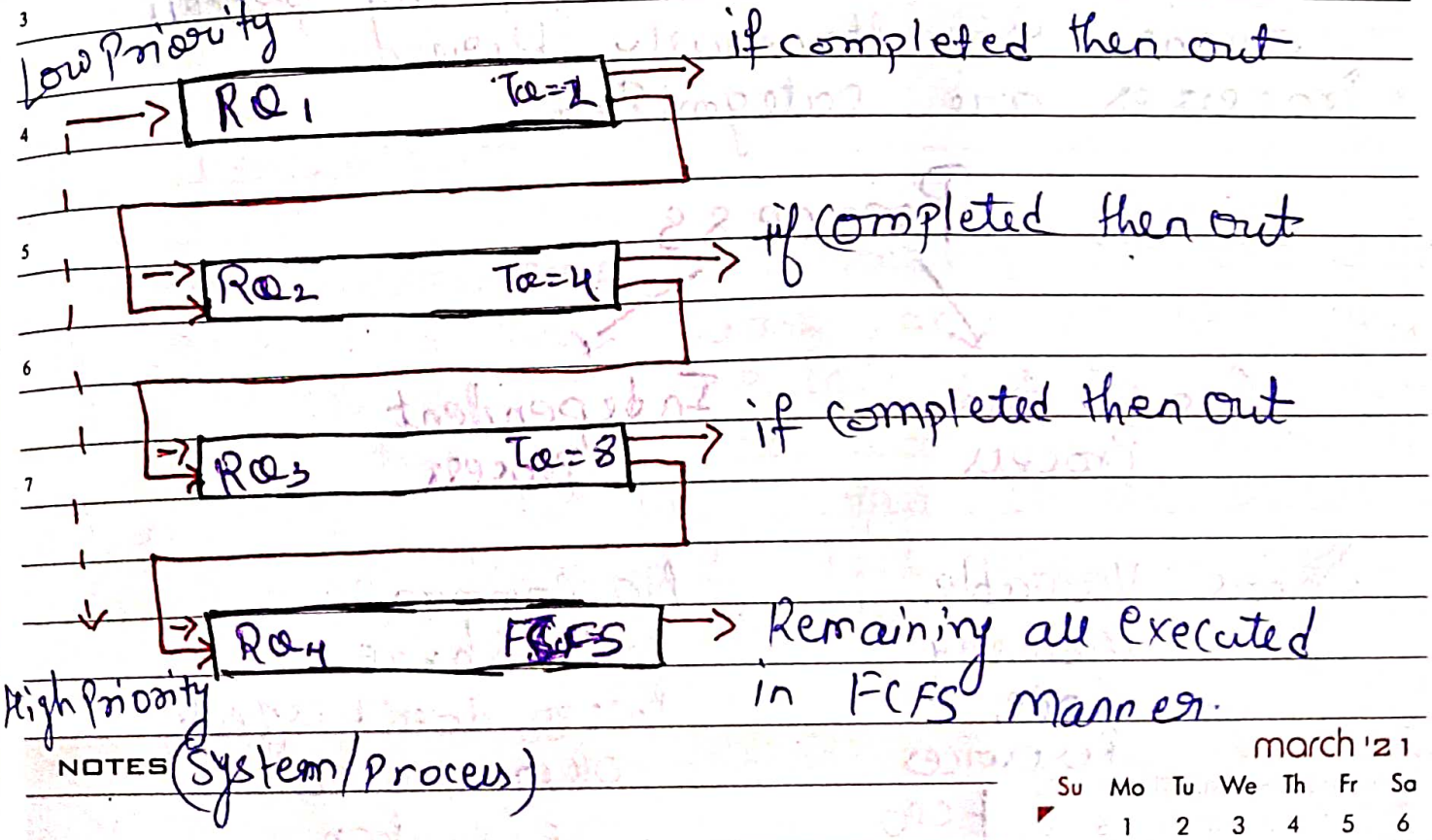
SU	MO	TU	WE	TH	FR	SA
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

NOTES

But here, some processes may starve for CPU if some higher priority queues are never becoming empty.
To solve this multilevel feedback queue is used.

Multilevel Feedback Queue Scheduling

Multilevel feedback queue, allows a process to move between queues. It is used to overcome the starvation during multilevel queue.



NOTES