Operating System

FCFS: CPU scheduling → Non-preemptive (once processor gets CPU, it doesn't release until it finishes its work)

Drawback: Convoy effect (makes small duration processor wait for long time)

Process	Arrival time	Burst time
P_1	0	4
P_2	1	3
P_3	2	1
P_4	3	2
P_5	4	5

⇒ Gantt chart

P_1	P_2	P_3	P_4	P_5

0 4 7 8 10 15

Process	Waiting time (startingtime - arrival time)	Turn around time
P_1	0 − 0 = 0	4 − 0 = 4
P_2	4 − 1 = 03	7 − 1 = 6
P_3	7 − 2 = 5	8 − 2 = 6
P_4	8 − 3 = 5	10 − 3 = 7
P_5	10 − 4 = 6	15 − 4 = 11

$$\text{Avg. waiting time} = \frac{0+3+5+5+6}{5}$$

$$= 3.8 \text{ m.sec}$$

Avg. Turn around time = $\frac{\cancel{4+7+8+10+15}}{5}$

~~= 8.8 msec~~

$$= \frac{4+6+6+7+11}{5}$$

$$= 6.8 \text{ msec}$$

Ex:

Process	Arrival time	Burst time
P_1	2	6
P_2	5	3
P_3	1	8
P_4	0	3
P_5	4	4

⇒ Gantt chart

P_4	P_3	P_1	P_5	P_2

0 3 11 17 21 24

Process	Waiting time	Turn around time
P_1	$11-2=9$ ~~0-0=0~~	$17-2=15$
P_2	$21-5=16$	$24-5=19$
P_3	$3-1=2$	$11-1=10$
P_4	$0-0=0$	$3-0=3$
P_5	$17-4=13$	$21-4=17$

Avg. waiting time $= \dfrac{9+16+2+0+13}{5}$

$= 8$ msec

Avg. turn around time = ~~$\frac{17+24+11+3+21}{5}$~~

$= \dfrac{15+19+10+3+17}{5}$ ~~= 15.2 msec.~~

$= 12.8$ msec.

Advantage of FCFS: Easy and simple to implement.

Round robin (RR) scheduling : Pre-emptive version of FCFS scheduling.
time will be divided into small slice.

Eg:

Process	Arrival time	Service time (burst time)
1	0	3
2	2	6
3	4	4
4	6	5
5	8	2

Time quantum or time slice = 1 msec.

⇒ Gantt chart.

1	1	2	1	2	3	2	4	3	2	5	4	3	2	5	4	3	2	4	4

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

Time	Head, Readylist, Tail.			
0	1	—	—	⇒ 1
1	1	—	—	⇒ 1
2	—	2	1	⇒ 2 1
3	1	—	2	⇒ 1 2
4	2	3	—	⇒ 2 3
5	3	—	2	⇒ 3 2
6	2	4	3	⇒ 2 4 3
7	4 3	—	~~3~~ 2	⇒ 4 3 2
8	3 2	5	4	⇒ 3 2 5 4
9	2 5 4	—	3	⇒ 2 5 4 3
10	5 4 3	—	2	⇒ 5 4 3 2
11	4 3 2	—	5	⇒ 4 3 2 5
12	3 2 5	—	4	⇒ 3 2 5 4
13	2 5 4	—	3	⇒ 2 5 4 3
14	5 4 3	—	2	⇒ 5 4 3 2
15	4 3 2	—	—	⇒ 4 3 2
16	3 2	—	4	⇒ 3 2 4
17	2 4	—	—	⇒ 2 4
18	4	—	—	⇒ 4
19	4	—	—	⇒ 4

Process	Waiting time	turn around time.
	last is left value, No. of [illegible] came before, arrival time	
1.	3 - 2 - 0 = 1	4 - 0 = 4
2	17 - 5 - 2 = 10.	18 - 2 = 16
3.	16 - 3 - 4 = 9	17 - 4 = 13
4	19 - 4 - 6 = 9	20 - 6 = 14
5	14 - 1 - 8 = 5	15 - 8 = 7

$$\text{Avg. waiting time} = \frac{1+10+9+9+5}{5}$$

$$= 6.8 \text{ msec.}$$

$$\text{Avg. turn around time} = \frac{4+16+13+14+7}{5}$$

$$= 10.8 \text{ msec.}$$

Ex:

Process	Arrival time	service time.
1	0	3
2	2	6
3	4	4
4	6	5
5	8	2

Time quantum = 2 msec.

⇒

Time	Head, Readylist, tail	
0	1 – –	⇒ 1
2	– 2 1	⇒ 21
4	1 3 2	⇒ 132
6	324 1	⇒ 3241
8	241 5 3	⇒ 24153
10	4153 – 2	⇒ 41532
12	1532 – 4	⇒ 15324
14	5324 – –	⇒ 5324
16	324 – 5	⇒ 3245.
18	245 – 3	⇒ 2453
20	453 – 2	⇒ 4532

~~532—4 ⇒ 5324~~

Time	Head, Ready List, tail	
0,1	1 — —	⇒ 1
2,3	— 2 1	⇒ 21
4	1 3 2	
5,6	3 2	

Time	Head, Ready List, tail	
0	1 — —	⇒ 1
1	1 — —	⇒ 1
2	— 2 1	⇒ 21
3	— 2 1	⇒ 21
4	1 3 2	⇒ 132
5	32 — —	⇒ 32
6	32 4 —	⇒ 324
7	2 4 — 3	⇒ 243
8	2435 —	⇒ 2435
9	435 — 2	⇒ 4352
10	435 — 2	⇒ 4352
11	3524	⇒ 3524
12	3524	⇒ 3524.
13	524-~~3~~	⇒ 524~~3~~
14	524-~~3~~	⇒ 524~~3~~.
15	24 — —	⇒ 24
16	24 — —	⇒ 24
17	4 — —	⇒ 4
18	4 — —	⇒ 4
19	4 — —	⇒ 4.

~~Process waiting time turn~~

Gantt chart:

1	1	2	2	1	3	3	2	2	4	4	3	3	5	5	2	2	4	4	4

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

Process	waiting time	turn around time
1	4-2-0=2	5-0=5
2	16-5-2=9	17-2=15
3	12-3-4=5	13-4=9
4	19-4-6=9	20-6=14
5	14-1-8=5	15-8=7

Avg. waiting time $= \frac{2+9+5+9+5}{5}$

$= 6$ msec.

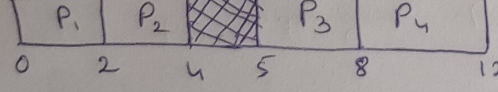
Avg. turn around time $= \frac{5+15+9+14+7}{5}$

$= 10$ msec.

<u>Ex:</u> FCFS.

Process	Arrival time	Burst time.
P_1	0	2
P_2	1	2
P_3	5	3
P_4.	6	4.

⇒ Gantt chart.



Process	waiting time	turn around time
P_1	0-0=0	2-0=2
P_2	2-01=1	4-1=3
P_3	5-5=0	8-5=3
P_4.	8-6=2	12-6=6

Avg. waiting time $= \frac{0+1+0+2}{4} = 0.75$ msec

Avg. turn around time $= \frac{2+3+3+6}{4} = 3.5$ msec.

Ex: FCFS.

Process	Burst time
P_1	10
P_2	1
P_3	2
P_4	1
P_5	5

$\Rightarrow$

P_1	P_2	P_3	P_4	P_5	
0	10	11	13	14	19

process	waiting time	Turn around ~~Burst~~ time
P_1	$0-0=0$	$10-0=10$
P_2	$10-0=10$	$11-0=11$
P_3	$11-0=11$	$13-0=13$
P_4	$13-0=13$	$14-0=14$
P_5	$14-0=14$	$19-0=19$

$$\text{Avg waiting time} = \frac{0+10+11+13+14}{5}$$

$$= 9.6 \text{ msec}$$

$$\text{Avg Turn around time} = \frac{10+11+13+14+19}{5}$$

$$= 13.4 \text{ msec}$$

Shortest Job First (SJF) scheduling: Non-preemptive ~~[illegible]~~

Shortest burst time processor uses the CPU first.

Less waiting time compared to FCFS.

Process	Burst Time	Turn Around time	Waiting time
P_1	6	9	3
P_2	8	24	16
P_3	7	16	9
P_4	3	3	0
		Avg = 13.	Avg = 7.

P_4	P_1	P_3	P_2	
0	3	9	16	24

Pre emptive SJF : SRTF (Shortest remaining time first).

Process	Arrival time	Burst time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5.

P_1	P_2	P_2	P_2	P_2	P_4	P_1	P_3	
0	1	2	3	4	5	10	17	26

Process	Turn around time	waiting time.
P_1	$17-0=17$	$10-1-0=9$.
P_2	$5-1=4$	$4-3-1=0$
P_3	$26-2=24$	$17-0-2=15$
P_4	$10-3=7$	$5-0-3=2$
	Avg = 13	Avg = 6.5.

Drawback: We cannot predict whether the upcoming processor has less burst time or not.

Priority scheduling:

Process	Burst time	Priority
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2.

P_2	P_5	P_1	P_3	P_4	
0	1	6	16	18	19

If two processors have same priority order, it works on First come first basis. If the arrival times are also same, the processor with less subscript gets higher priority.

Process	Turn Around time	Waiting time.
P_1	16	6
P_2	1	0
P_3	18	16
P_4	19	18
P_5.	6	1
	Avg = 12	Avg = 8.2.

8.

Process	Burst ~~Arrival~~ time.	Priority.
P_1	10	3
P_2	1	1
P_3	2	3
P_4	1	4
P_5.	5	2

$P_1 > P_3$ lower subscript hight priority.

P_2	P_5	P_1	P_3	P_4

0 1 6 16 18 19

Process	TAT	WT.
P_1	16	6
P_2	1	0
P_3	18	16
P_4	19	18
P_5.	6	1
	Avg=12	Avg =8.2.

9.

Process	Burst time	Priority
P_1	2	2
P_2	1	1
P_3	8	4
P_4	4	2
P_5	5	3

i) FCFS
ii) Non preemptive SJF
iii) Non Preemptive priority (larger no. high priority)
iv) RR (Q=2)

FCFS

i)

P_1	P_2	P_3	P_4	P_5	
0	2	3	11	15	20

Process	TAT	WT
P_1	2	0
P_2	3	2
P_3	11	3
P_4	15	11
P_5	20	15
	Avg = 10.2	6.2

ii) Non-premptive SJF

P_2	P_1	P_4	P_5	P_3	
0	1	3	7	12	20

Process	TAT	WT
P_1	3	1
P_2	1	0
P_3	20	12
P_4	7	3
P_5	12	7
	Avg = 8.6	Avg = 4.6

iii) Non Premptive Priority.

P3	P5	P1	P4	P2

0 8 13 15 19 20.

Process	TAT	WT.
P_1	15	13
P_2	20	19
P_3	8	0
P_4	19	15
P_5.	13	8
	Avg = 15	Avg = 11

iv) RR (Q = 2).

P1	P2	P3	P4	P5	P3	P4	P5	P3	P5	P3

0 2 3 5 7 9 11 13 15 17 18 20.

Process	TAT	WT
P_1	2	0
P_2	3	2
P_3	20	19 - 7 = 12
P_4	13	12 - 3 = 9
P_5.	18.	17 - 4 = 13
	Avg = 11.2	Avg = 7.2

Page replacement algorithms in virtual memory:

1. FIFO: oldest page is replaced.

Reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

Page frame = 4

7	0	1	2	3	4	0	1	2	7
7	7	7	7	3	3	3	3	2	2
	0	0	0	0	4	4	4	4	7
		1	1	1	1	0	0	0	0
			2	2	2	2	1	1	1

Page faults = 10

Ex: Reference string:

1 2 3 4 1 2 5 1 2 3 4 5

3 frames:

1	1	1	4	4	4	5	5	5
	2	2	2	1	1	1	3	3
		3	3	3	2	2	2	4

Page faults = 9

4 frames:

1	1	1	1	5	5	5	5	4	4
	2	2	2	2	1	1	1	1	5
		3	3	3	3	2	2	2	2
			4	4	4	4	3	3	3

page faults = 10.

With increase in frame the page faults should decrease but in some problems because of Belady's Anomaly, page faults also increase.

Drawback: Belady's Anomaly.

2. Optimal Algorithm:

Reference string:

7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

7	7	7	7	3	3	1	1
	0	0	0	0	0	0	0
		1	1	1	4	4	7
			2	2	2	2	2

Page faults = 8

3. Least Recently used (LRU) Algorithm:

7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

7	7	7	7	3	3	3	7
	0	0	0	0	0	0	0
		1	1	1	4	1	1
			2	2	2	2	2

Page faults = 8

Deadlock:

Banker's algorithm: Used to avoid deadlock.

Q. Consider the following snapshot of a system

	Allocation	Max	Available
	A B C	A B C	A B C
P_0	0 1 0	7 5 3	3 3 2
P_1	2 0 0	3 2 2	
P_2	3 0 2	9 0 2	
P_3	2 1 1	2 2 2	
P_4	0 0 2	4 3 3	

Total: A = 10 instances of resource A are available

B = 5

C = 7

⇒

	Allocation	Max	Available (Total - Allocation)	Remaining Need (Max - Allocation)	Safe sequence
	A B C	A B C	A B C	A B C	
P_0	0 1 0 ④	7 5 3	3 3 2	7 4 3 ≤ 745 ✓	P_1
P_1	2 0 0 ①	3 2 2	+ 2 0 0 = 5 3 2	1 2 2 ≤ 332 ✓	↓ P_3
P_2	3 0 2 ⑤	9 0 2	+ 2 1 1 = 7 4 3	6 0 0 ≤ 755 ✓	↓ P_4
P_3	2 1 1 ②	2 2 2	+ 0 0 2 = 7 4 5	0 1 1 ≤ 532 ✓	↓ P_0
P_4	0 0 2 ③	4 3 3	+ 0 1 0 = 7 5 5	4 3 1 ≤ 743 ✓	↓ P_2
			+ 3 0 2 = 10 5 7		

∴ The system is safe

Q. Continuation

Suppose that process P_1 requests one instance of resource type A and two instances of resource type C, can this request be granted immediately. (1 0 2)

⇒

	Allocation A B C	Max A B C	Available A B C	Remaining Need A B C
			3 3 2	
			− 1 0 2	
P_0	0 1 0 (4)	7 5 3	2 3 0	7 4 3 ✓
			+ 3 0 2	
P_1	3 0 2 (1)	3 2 2	5 3 2	0 2 0 ✓
			+ 2 1 1	
P_2	3 0 2 (5)	9 0 2	7 4 3	6 0 0 ✓
			+ 0 0 2	
P_3	2 1 1 (2)	2 2 2	7 4 5	0 1 1 ✓
			+ 0 1 0	
P_4	0 0 2 (3)	4 3 3	7 5 5	4 3 1 ✓
			+ 3 0 2	
			10 5 7	

$P_1 \rightarrow P_3 \rightarrow P_4 \rightarrow P_0 \rightarrow P_2$

∴ System is safe.

Q. Request 2 by Process P_4 = 3, 3, 0

⇒ The request 2 cannot be granted since, the resources are not available. (2 3 0 $\ngtr$ 3 3 0)

Q. Request 3 by process P_0 = 0, 2, 0.

	Allocation A B C	Max A B C	Available A B C	Remaining Need A B C
P_0	(0 1 0 + 0 2 0) 0 3 0	7 5 3	(2 3 0 − 0 2 0) 2 1 0	7 2 3 ✗
P_1	3 0 2	3 2 2	~~5 1 2~~	0 2 0 ✗
P_2	3 0 2	9 0 2	~~7 2 3~~	6 0 0 ✗
P_3	2 1 1	2 2 2	~~7 2 5~~	0 1 1 ✗
P_4	0 0 2	4 3 3	~~7 5 5~~	4 3 1 ✗
			~~10 5 7~~	

~~$P_1 \rightarrow P_3 \rightarrow P_4 \rightarrow P_0 \rightarrow P_2$~~

~~∴ System is safe.~~

System is not safe.

Request cannot be granted.