Q. Consider the following set of processes:

Process	Arrival time(ms)	Burst time(ms)
A	3	4
B	5	3
C	0	2
D	5	1
E	4	3

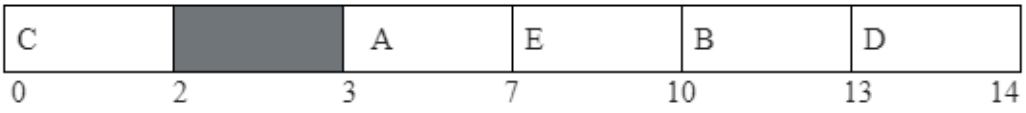
Using FCFS scheduling,draw the Gantt chart and find out the following:

i) Average waiting time

ii)Average turnaround time.

Solution:

Gantt chart:



Process	Arrival time(AT) (ms)	Burst time(BT) (ms)	Turnaround time(TAT) (Exit time-Arrival time) (ms)	Waiting time (TAT-BT) (ms)
A	3	4	7-3=4	4-4=0
B	5	3	13-5=8	8-3=5
C	0	2	2-0=2	2-2=0
D	5	1	14-5=9	9-1=8
E	4	3	10-4=6	6-3=3

i)Average waiting time=(0+5+0+8+3)/5 =3.2 ms

ii)Average turnaround time=(4+8+2+9+6)/5=5.8 ms

Note: Always non-preemptive in nature.

Q. Consider the following set of processes:

Process	Arrival Time(AT)	Burst time(BT)
P1	2	6
P2	5	2
P3	1	8
P4	0	3
P5	4	4

Using SJF scheduling(Non-preemptive),draw the gantt chart and find out the following:

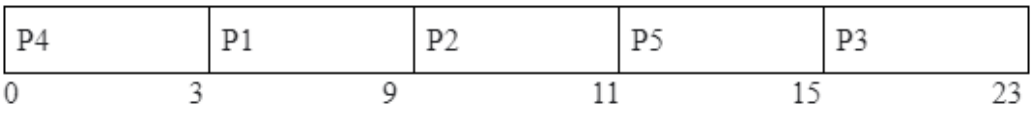
i)Average waiting time

ii)Average response time

iii)Average turnaround time

Solution: Criteria:Burst time, Mode: Non-preemptive

Gantt chart:



Process	Arrival Time(AT)	Burst time(BT)	Completion time(CT)	Turnaround time (TAT)= (CT-AT)	Waiting time (WT)= (TAT-BT)	Response time(RT)
P1	2	6	9	7	1	1
P2	5	2	11	6	4	4
P3	1	8	23	22	14	14
P4	0	3	3	3	0	0
P5	4	4	15	11	7	7

i)Average waiting time=(1+4+14+0+7)/5=5.2 unit

ii)Average response time=(1+4+14+0+7)/5=5.2 unit

iii)Average turnaround time=(7+6+22+3+11)/5=9.8 unit

Q. Consider the following set of processes:

Process	Arrival Time(AT)	Burst time(BT)
P1	0	5
P2	1	3
P3	2	4
P4	4	1

Using SJF scheduling(preemptive),draw the gantt chart and find out the following:

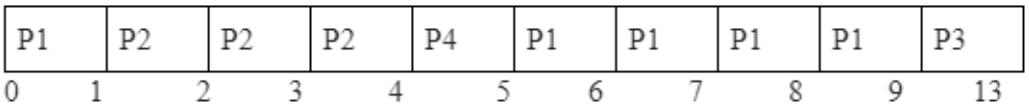
i)Average waiting time

ii)Average response time

iii)Average turnaround time

Solution:

Gantt chart:



Process	Arrival Time(AT)	Burst time(BT)	Completion time(CT)	Turnaround time (TAT)= (CT-AT)	Waiting time (WT)= (TAT-BT)	Response time(RT)=(CPU first time-AT)
P1	0	5	9	9	4	0
P2	1	3	4	3	0	0
P3	2	4	13	11	7	7
P4	4	1	5	1	0	0

i)Average waiting time=(4+0+7+0)/4=2.75 unit

ii)Average response time=(0+0+7+0)/4=1.75 unit

iii)Average turnaround time=(9+3+11+1)/4=6 unit

Q. Consider the following set of processes:

Process	Arrival Time(AT)	Burst time(BT)
P1	0	5
P2	1	3
P3	2	1
P4	3	2
P5	4	3

Using Round Robin scheduling(Assume time quantum=2),draw the gantt chart and find out the following:

i)Average waiting time

ii)Average response time

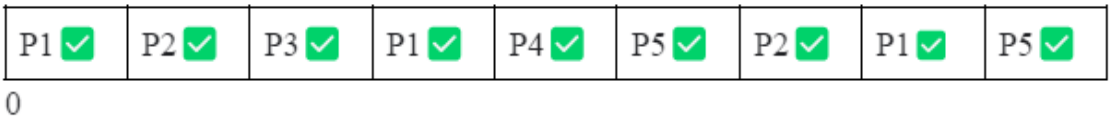
iii)Average turnaround time

Solution:

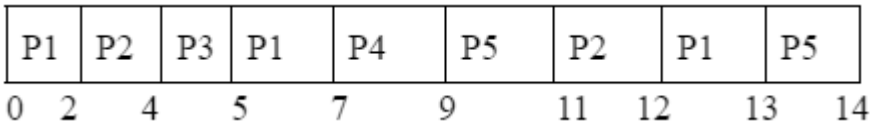
Criteria: Time quantum

Mode:Preemptive

Ready Queue:



Running Queue/Gantt chart



Process	Arrival Time(AT)	Burst time(BT)	Completion time(CT)	Turnaround time (TAT)= (CT-AT)	Waiting time (WT)= (TAT-BT)	Response time (RT=(CPU first time-AT)
P1	0	5-3-2-0	13	13	8	0
P2	1	3-1-0	12	11	8	1
P3	2	1-0	5	3	2	2
P4	3	2-0	9	6	4	4
P5	4	3-1-0	14	10	7	5

i)Average waiting time=(8+8+2+4+7)/5=5.8 unit

ii)Average response time=(0+1+2+4+5)/5=2.4 unit

iii)Average turnaround time=(13+11+3+6+10)/5=8.6 unit

Q. Consider the following set of processes:

Process	**Arrival Time(AT)**	**Burst time(BT)**	**Priority**
P1	**0**	**4**	**2**
P2	**1**	**3**	**3**
P3	**2**	**1**	**4**
P4	**3**	**5**	**5**
P5	**4**	**2**	**5**

Using Priority scheduling(non-preemptive and preemptive),draw the gantt chart and find out the following:

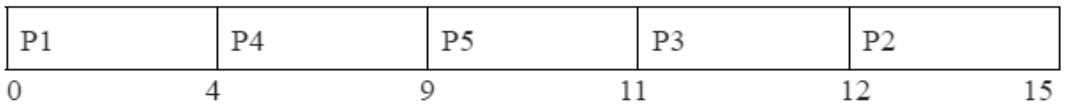
i)Average waiting time

ii)Average turnaround time

Note:Higher number represents higher priority

Solution: Mode: Non-preemptive

Gantt chart:



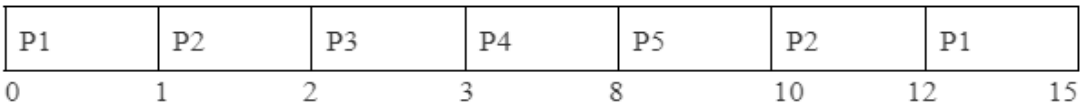
Process	Arrival time (AT)	Burst time (BT)	Turnaround time (TAT) (Exit time-Arrival time)	Waiting time (TAT-BT)
P1	0	4	4-0=4	4-4=0
P2	1	3	15-1=14	14-3=11
P3	2	1	12-2=10	10-1=9
P4	3	5	9-3=6	6-5=1
P5	4	2	11-4=7	7-2=5

i) Average waiting time = (0 + 11 + 9 + 1 + 5) / 5 = 26 / 5 = 5.2 unit

ii) Average Turnaround time = (4 + 14 + 10 + 6 + 7) / 5 = 41 / 5 = 8.2 unit

Mode: Preemptive

Gantt chart:



Process	Arrival time (AT)	Burst time (BT)	Turnaround time (TAT) (Exit time-Arrival time)	Waiting time (TAT-BT)
P1	0	4	15-0=15	15-4=11
P2	1	3	12-1=11	11-3=8
P3	2	1	3-2=1	1-1=0
P4	3	5	8-3=5	5-5=0
P5	4	2	10-4=6	6-2=4

i)Average waiting time = (11 + 8 + 0 + 0 + 4) / 5 = 23 / 5 = 4.6 unit

Bankers Algorithm Numericals

Q. Consider a system that contains five processes P1, P2, P3, P4, P5 and the three resource types A, B and C. Following are the resource types: A has 10, B has 5 and the resource type C has 7 instances.

Processes	Allocation	Max	Available
	A B C	A B C	A B C
P1	0 1 0	7 5 3	3 3 2
P2	2 0 0	3 2 2	
P3	3 0 2	9 0 2	
P4	2 1 1	2 2 2	
P5	0 0 2	4 3 3	

Answer the following questions using the banker's algorithm:

1. **What is the reference of the need matrix?**
2. **Determine if the system is safe or not.**
3. **What will happen if the resource request (1, 0, 0) for process P1 can the system accept this request immediately?**

Ans:

1. Context of the need matrix is as follows:

Need [i] = Max [i] - Allocation [i]

Need for P1: (7, 5, 3) - (0, 1, 0) = 7, 4, 3

Need for P2: (3, 2, 2) - (2, 0, 0) = 1, 2, 2

Need for P3: (9, 0, 2) - (3, 0, 2) = 6, 0, 0

Need for P4: (2, 2, 2) - (2, 1, 1) = 0, 1, 1

Need for P5: (4, 3, 3) - (0, 0, 2) = 4, 3, 1

Process	Need
	A B C
P1	7 4 3
P2	1 2 2
P3	6 0 0
P4	0 1 1
P5	4 3 1

2.Apply the Banker's Algorithm:

Available Resources of A, B and C are 3, 3, and 2.

Now we check if each type of resource request is available for each process.

Step 1: For Process P1:

Need <= Available

7, 4, 3 <= 3, 3, 2 condition is false.

So, we examine another process, P2

Step 2: For Process P2:

Need <= Available

1, 2, 2 <= 3, 3, 2 condition true

New available = available + Allocation

(3, 3, 2) + (2, 0, 0) => 5, 3, 2

Similarly, we examine another process P3.

Step 3: For Process P3:

P3 Need <= Available

6, 0, 0 < = 5, 3, 2 condition is false.

Similarly, we examine another process, P4.

Step 4: For Process P4:

P4 Need <= Available

0, 1, 1 <= 5, 3, 2 condition is true

New Available resource = Available + Allocation

5, 3, 2 + 2, 1, 1 => 7, 4, 3

Similarly, we examine another process P5.

Step 5: For Process P5:

P5 Need <= Available

4, 3, 1 <= 7, 4, 3 condition is true

New available resource = Available + Allocation

7, 4, 3 + 0, 0, 2 => 7, 4, 5

Now, we again examine each type of resource request for processes P1 and P3.

Step 6: For Process P1:

P1 Need <= Available

7, 4, 3 <= 7, 4, 5 condition is true

New Available Resource = Available + Allocation

7, 4, 5 + 0, 1, 0 => 7, 5, 5

So, we examine another process P2.

Step 7: For Process P3:

P3 Need <= Available

6, 0, 0 <= 7, 5, 5 condition is true

New Available Resource = Available + Allocation

7, 5, 5 + 3, 0, 2 => 10, 5, 7

Hence, we execute the banker's algorithm to find the safe state and the safe sequence like P2, P4, P5, P1 and P3.

3. For granting the Request (1, 0, 2), first we have to check that Request <= Available, that is (1, 0, 2) <= (3, 3, 2), since the condition is true. So the process P1 gets the request immediately.

Page Replacement Policy + Numericals on it

Q. Explain Page Replacement Policy ?

Ans :

1. Page replacement policy is a crucial aspect of memory management in operating systems, specifically in the context of virtual memory.

2. Virtual memory allows the execution of processes that are larger than the physical memory available by using disk space as an extension.

3. Pages are fixed-sized blocks of memory used in virtual memory systems, and the main memory is divided into page frames of the same size.

4. When a process requests a page that is not currently in the main memory, a page fault occurs.

5. Page replacement policy determines which page should be evicted from the main memory to make space for the incoming page.

6. The goal of a page replacement policy is to minimize the number of page faults and optimize memory utilization.

7. There are various page replacement policies, including:

   a. Least Recently Used (LRU): This policy replaces the page that has not been used for the longest period. It assumes that pages that were recently used are more likely to be used again in the near future.

   b. First-In, First-Out (FIFO): This policy replaces the page that has been in the main memory the longest. It follows the principle of a queue, where the oldest page is the first to be replaced.

   c. Optimal: This is an idealized page replacement policy that evicts the page that will not be used for the longest duration in the future. However, it requires knowledge of future memory references, which is usually not feasible.

   d. Clock (or Second-Chance): This policy maintains a circular list of pages in the main memory and uses a "use" bit associated with each page. When a page needs to be replaced, it checks the use bit and gives pages a second chance if they have been recently referenced.

8. The choice of page replacement policy depends on factors such as the system's workload, memory size, and performance requirements.

9. The performance of a page replacement policy is often evaluated based on metrics like page fault rate, hit rate, and overall system efficiency.

Numericals in Page Replacement Policy

Q. Consider a reference string: 4, 7, 6, 1, 7, 6, 1, 2, 7, 2. the number of frames in the memory is 3. Find out the number of page faults respective to:

1. Optimal Page Replacement Algorithm
2. FIFO Page Replacement Algorithm
3. LRU Page Replacement Algorithm

Optimal Page Replacement Algorithm

Request	4	7	6	1	7	6	1	2	7	2
Frame 3			6	6	6	6	6	2	2	2
Frame 2		7	7	7	7	7	7	7	7	7
Frame 1	4	4	4	1	1	1	1	1	1	1
Miss/Hit	Miss	Miss	Miss	Miss	Hit	Hit	Hit	Miss	Hit	Hit

Number of Page Faults in Optimal Page Replacement Algorithm = 5

LRU Page Replacement Algorithm

Request	4	7	6	1	7	6	1	2	7	2
Frame 3			6	6	6	6	6	2	2	2
Frame 2		7	7	7	7	7	7	7	7	7
Frame 1	4	4	4	1	1	1	1	1	1	1
Miss/Hit	Miss	Miss	Miss	Miss	Hit	Hit	Hit	Miss	Hit	Hit

Number of Page Faults in LRU = 6

FIFO Page Replacement Algorithm

Request	4	7	6	1	7	6	1	2	7	2
Frame 3			6	6	6	6	6	2	2	2
Frame 2		7	7	7	7	7	7	7	7	7
Frame 1	4	4	4	1	1	1	1	1	1	1
Miss/Hit	Miss	Miss	Miss	Miss	Hit	Hit	Hit	Miss	Hit	Hit

Number of Page Faults in FIFO = 6

Q. Consider the following page reference string:

1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6.

How many page faults occur for the following page replacement algorithms assuming one, three, four frames? Assume that all pages are initially used by pages left over from another process, so first unique pages will all cost one fault each.

Solution: **LRU:**

3 frames

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
Frame1	1			4			5			1			7			2		-		
Frame2		2			-			6				3			-				-	
Frame3			3			1			2		-			6			1			6

15 faults

4 frames:

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
Frame1	1					-				-				6						-
Frame2		2			-				-		-					-		-		
Frame3			3				5					3			-				-	
Frame4				4				6					7				1			

10 faults.

FIFO:

3 frames:

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
Frame1	1			4				6			-	3			-	2				6
Frame2		2			-	1			2				7				1			
Frame3			3				5			1				6				-	3	

16 faults

4 frames:

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
Frame1	1					-	5					3			-		1			
Frame2		2			-			6					7						3	
Frame3			3						2		-			6						-
Frame4				4						1						2		-		

14 faults.

Optimal:

3 frames:

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
Frame1	1					-				-		3			-				-	
Frame2		2			-				-		-		7			2		-		
Frame3			3	4			5	6						-			1			6

11 faults.

4 frames:

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
Frame1	1					-				-			7				1			
Frame2		2			-				-		-					-		-		
Frame3			3									-			-				-	
Frame4				4			5	6						-						-

8 faults.