



SRM Institute of Science and Technology
College of Engineering and Technology
School of Computing
SRM Nagar, Kattankulathur – 603203, Chengalpattu District, Tamilnadu
Academic Year: 2024-25 (ODD)

SET A

Test: FJ2

Date: 01.10.2024

Course Code & Title: 21CSC202J - Operating Systems

Duration: 100 Minutes

Year & Sem: II Year / III Sem

Max. Marks: 50

Course Articulation Matrix: (to be placed)

S.No	Course Outcome	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
1	CO1	3	3	2	2								3			
2	CO2	3	3	3	2								3			
3	CO3	3	3	3	2								3			
4	CO4	3	3	3	2								3			
5	CO5	3	2	3	2								3			

Questions with Answer Key

Part – A (10 x 1 = 10 Marks)						
Instructions: Answer all						
Q. No	Answer with choice variable	Marks	BL	CO	PO	PI Code
1	Systems that permit only a single process to execute at any given time are referred to as _____ a) uniprogramming systems b) uniprocessing systems c) unitasking systems d) multiprocessing systems Answer: b uniprocessing systems	1	L1	2	2	1.2.2
2	The processes that are residing in main memory and are ready and waiting to execute are kept on a list called _____ a) job queue b) ready queue c) device queue d) process queue Answer : b ready queue	1	L1	2	2	1.5.1

3	<p>----- Scheduler selects which process should be executed next and allocates CPU</p> <p>a. DMA controller</p> <p>b. CPU Scheduler</p> <p>c. Short Term Scheduler</p> <p>d. Long Term Scheduler</p> <p>Answer : c Short Term Scheduler</p>	1	L2	2	3	1.6.1
4	<p>.-----refers to the mechanisms that allow processes to communicate and coordinate their actions when running concurrently in an operating system</p> <p>a. Inter-process communication</p> <p>b. Inter-system communication</p> <p>c. Message communication</p> <p>d. Synchornization</p> <p>Answer : a Inter-process communication</p>	1	L2	2	2	1.5.1
5	<p>When one thread immediately terminates the target thread, it is called _____</p> <p>a) Asynchronous cancellation</p> <p>b) Systematic cancellation</p> <p>c) Sudden Termination</p> <p>d) Deferred cancellation</p> <p>Ansewer: a Asynchronous cancellation</p>	1	L2	2	3	1.6.1
6	<p>A process is advanced to the ready state in a preemptive scheduling algorithm when:</p> <p>a) A new process arrives</p> <p>b) A process terminates</p> <p>c) A process switches from running to waiting</p> <p>d) A process switches from waiting to ready</p> <p>Ans: a) A new process arrives</p>	1	2	3	2	2.5.1
7	<p>What is the main difference between soft real-time systems and hard real-time systems?</p> <p>A) Soft real-time systems can miss deadlines occasionally without severe consequences, whereas hard real-time systems must meet all deadlines.</p> <p>B) Hard real-time systems provide no guarantee as to when a critical</p>	1	2	3	1	1.6.1

	real-time process will be scheduled, while soft real-time systems guarantee deadline adherence. C) Soft real-time systems guarantee that processes will be given preference over noncritical processes, whereas hard real-time systems guarantee deadline adherence. D) Soft real-time systems and hard real-time systems have no significant differences in scheduling. Answer: C) Soft real-time systems guarantee that processes will be given preference over noncritical processes, whereas hard real-time systems guarantee deadline adherence.						
8	In a system with multiple processes and resources, a deadlock detection algorithm primarily checks for _____ A. The maximum number of resources each process can request. B. Cycles in the resource allocation graph. C. The order of resource requests. D. The time taken by each process. Answer: B) Cycles in the resource allocation graph.	1	1	3	1	1.3.1	
9	Select the type of CPU Scheduling algorithm that has a large average waiting time? a) First come First Serve (FCFS) b) Priority based Scheduling Algorithm c) Round Robin Scheduling Algorithm (RR) d) Multilevel Feedback Queue Algorithm Answer:c) Round Robin Scheduling Algorithm (RR)	1	1	3	1	1.3.1	
10	Only the process executing the critical section is allowed to access the shared variable and all other processes should be prevented from doing so until the completion of the critical section. This is known as a) Inter-process Communication b) Semaphore c) Deadlock d) Mutual Exclusion Answer: d) Mutual Exclusion	1	1	1	3L2	3	1.2.1
Part – B (4 x 5 = 20 Marks)							
11	What are the states undergone by a process when a user opens more than one application in a laptop at the same time? Illustrate the same using suitable diagrams. Answer: As a process executes, it changes state new: The process is being created running: Instructions are being executed waiting: The process is waiting for some event to occur ready: The process is waiting to be assigned to a processor terminated: The process has finished execution	5	L2	2	2	2.6.2	

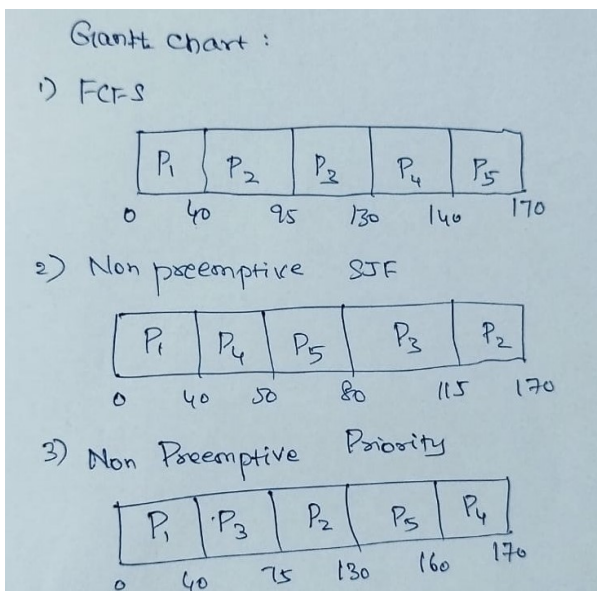
	<pre> graph TD new((new)) -- admitted --> ready((ready)) ready -- interrupt --> ready ready -- scheduler dispatch --> running((running)) running -- I/O or event completion --> ready running -- I/O or event wait --> running running -- exit --> terminated((terminated)) waiting((waiting)) -- I/O or event completion --> ready </pre>					
12	<p>Explain the role of semaphore in achieving mutual exclusion.</p> <p>Answer:</p> <p>A semaphore is a more general synchronization primitive than a mutex, capable of controlling access to a resource pool. A binary semaphore (also called a mutex semaphore) functions similarly to a mutex, while a counting semaphore can allow multiple threads to access a finite number of instances of a resource.</p> <p>Basic Steps:</p> <ol style="list-style-type: none"> 1. Initialize the Semaphore: Set the initial count of the semaphore. 2. Wait (P) Operation: Decrement the semaphore count. If the count is zero, the thread is blocked. 3. Signal (V) Operation: Increment the semaphore count. If the count is positive, a waiting thread is unblocked. <p>Advantages:</p> <ul style="list-style-type: none"> • Can handle more complex synchronization scenarios, such as allowing a limited number of threads to access a resource concurrently. • Flexible and powerful for various synchronization problems. <p>Disadvantages:</p> <ul style="list-style-type: none"> • More complex to use correctly compared to mutexes. • Prone to issues like deadlocks, priority inversion, and semaphore leaks if not managed properly. 	5	L2	2	2	2.6.2
13	<p>Enumerate the necessary conditions for a deadlock to occur.</p> <p>Answer:</p> <ol style="list-style-type: none"> 1. Mutual Exclusion: At least one resource must be held in a non-shareable mode. That is, only one process can use the resource at any given time. If another process requests that resource, the requesting process must be delayed until the resource has been released. 2. Hold and Wait: A process must be holding at least one resource and waiting to acquire additional resources that are currently being held by other processes. This means processes must be able to hold resources while waiting for others. 3. No Preemption: Resources cannot be forcibly taken from a process holding them. A resource can only be released voluntarily by the process holding it after that process has completed its task. 4. Circular Wait: There must be a circular chain of processes such that each process holds at least one resource needed by the next process in the chain. This implies that there is a cycle 	5	2	3	1	1.2.1

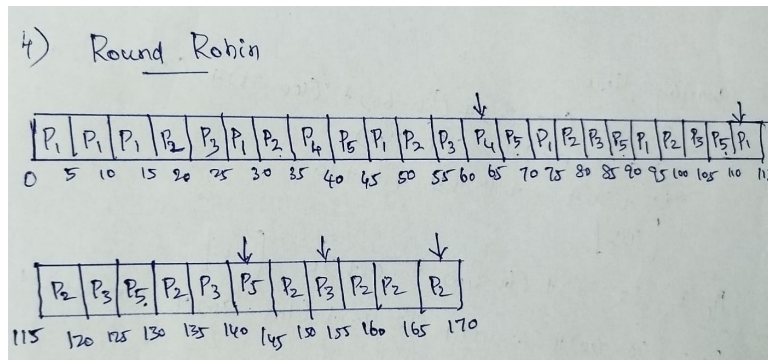
	in the resource allocation graph																				
14	<p>Consider four processes with the length of CPU burst time and arrival time. Create Gantt charts illustrating the execution of these processes using SRT.</p> <table> <tr> <th>Process</th> <th>Arrival Time</th> <th>Burst Time</th> </tr> <tr> <td>P_1</td> <td>0.0</td> <td>7</td> </tr> <tr> <td>P_2</td> <td>2.0</td> <td>4</td> </tr> <tr> <td>P_3</td> <td>4.0</td> <td>1</td> </tr> <tr> <td>P_4</td> <td>5.0</td> <td>4</td> </tr> </table> <p>Answer:</p> <p>Average waiting time</p> $= \frac{[(0-0) + (11-2)] + [(2-2) + (5-4)] + (4-4) + (7-5)}{4}$ $= \frac{9 + 1 + 0 + 2}{4}$ $= 3$ <p>Average turn-around time</p> $= \frac{(16-0) + (7-2) + (5-4) + (11-5)}{4}$ $= 7$	Process	Arrival Time	Burst Time	P_1	0.0	7	P_2	2.0	4	P_3	4.0	1	P_4	5.0	4	5	3	3	2	2.1.3
Process	Arrival Time	Burst Time																			
P_1	0.0	7																			
P_2	2.0	4																			
P_3	4.0	1																			
P_4	5.0	4																			
<p align="center">Part – C Either OR Choice Questions (2 X 10 = 20 Marks)</p>																					
15a	<p>Design and describe the activity of the process manager that handles the removal of the running process from the CPU and the selects another process based on any strategy with respect to processes scheduling. Represent the process scheduling using Queueing diagram.</p> <p>Answer:</p> <ol style="list-style-type: none"> The OS maintains all Process Control Blocks (PCBs) in Process Scheduling Queues. The OS maintains a separate queue for each of the process states and PCBs of all processes in the same execution state are placed in the same queue. When the state of a process is changed, its PCB is unlinked from its current queue and moved to its new state queue. Job queue – This queue keeps all the processes in the system. Ready queue – This queue keeps a set of all processes residing in main memory, ready and waiting to execute. A new process is always put in this queue. Device queues – The processes which are blocked due to unavailability of an I/O device constitute this queue. Schedulers are special system software which handle 	10	L4	2	3	2.6.3															

	<p>process scheduling in various ways. Their main task is to select the jobs to be submitted into the system and to decide which process to run. Schedulers are of three types</p> <ol style="list-style-type: none"> Long-Term Scheduler Short-Term Scheduler Medium-Term Scheduler 					
--	---	--	--	--	--	--

(or)

15b	<p>What is a race condition? Describe a solution to the Dining philosopher problem so that no races arise.</p> <p>Answer:</p> <p>Race condition: The situation where several processes access – and manipulate shared data concurrently. The final value of the shared data depends upon which process finishes last. To prevent race conditions, concurrent processes must be synchronized. Data consistency requires that only one processes should update the value of a data item at any time. This is ensured through the notion of a critical section.</p> <p>Dining philosopher solution:</p> <p>Wait and Signal operations will be used for the solution of the Dining Philosophers Problem, for picking a chopstick wait operation can be executed while for releasing a chopstick signal semaphore can be executed.</p> <p>Semaphore: A semaphore is an integer variable in S, that apart from initialization is accessed by only two standard atomic operations - wait and signal, whose definitions are as follows:</p> <pre> void Philosopher { while(1) { Wait(take_chopstickC[i]); Wait(take_chopstickC[(i+1) % 5]); </pre>	10	L4	2	3	2.6.3
-----	--	----	----	---	---	-------

	<pre>.. . EATING THE NOODLE Signal(put_chopstickC[i]); Signal(put_chopstickC[(i+1) % 5]); . . THINKING } }</pre>																				
16a	<p>Consider five processes with the length of CPU burst time, Arrival time and Priority (small value means high priority)</p> <table><thead><tr><th>Process</th><th>Burst Time</th><th>Arrival Time</th></tr></thead><tbody><tr><td>P1</td><td>40</td><td>0</td></tr><tr><td>P2</td><td>55</td><td>15</td></tr><tr><td>P3</td><td>35</td><td>15</td></tr><tr><td>P4</td><td>10</td><td>35</td></tr></tbody></table> <p>(i) Create Gantt charts illustrating the execution of these processes using FCFS, non preemptive SJF, priority scheduling and Round Robin scheduling (quantum = 5 ms) algorithms for this set of processes.</p> <p>(ii) Calculate the turnaround time for process P3 under each of the above algorithms.</p> <p>(iii) Calculate average waiting time for each algorithm and find out which algorithm would give the minimum average waiting time.</p> <p>Solution:</p> <p>i)</p> 	Process	Burst Time	Arrival Time	P1	40	0	P2	55	15	P3	35	15	P4	10	35	10	3	3	3	3.2.3
Process	Burst Time	Arrival Time																			
P1	40	0																			
P2	55	15																			
P3	35	15																			
P4	10	35																			



(ii) Turnaround Time for P3 under each algorithm

Turnaround time is calculated as the time from the process's arrival to its completion.

- **FCFS:** P3 arrives at time 15 and finishes at time 130.
 - Turnaround Time = $130 - 15 = 115$ ms
- **SJF:** P3 arrives at time 15 and finishes at time 115.
 - Turnaround Time = $115 - 15 = 100$ ms
- **Priority:** P3 arrives at time 40 and finishes at time 75.
 - Turnaround Time = $75 - 15 = 60$ ms
- **Round Robin:** P3 receives CPU time in slices. P3 arrives at time 15 and finishes at time 155.
 - Turnaround Time = $155 - 15 = 140$ ms

iii)

FCFS

- Waiting Time P1 = 0 ms (starts immediately)
- Waiting Time P2 = $40 - 15 = 25$ ms
- Waiting Time P3 = $95 - 15 = 80$ ms
- Waiting Time P4 = $130 - 35 = 95$ ms
- Waiting Time P5 = $140 - 40 = 100$ ms

Average Waiting Time (FCFS) = $(0 + 25 + 80 + 95 + 100) / 5$
= 60 ms

SJF

- Waiting Time P1 = 0 ms
- Waiting Time P2 = $115 - 15 = 100$ ms
- Waiting Time P3 = $80 - 15 = 65$ ms
- Waiting Time P4 = $40 - 35 = 5$ ms
- Waiting Time P5 = $50 - 40 = 10$ ms

Average Waiting Time (SJF) = $(0 + 100 + 65 + 5 + 10) / 5 = 36$ ms

Priority Scheduling

- Waiting Time P1 = 0 ms
- Waiting Time P2 = 75 - 15 = 60 ms
- Waiting Time P3 = 40 - 15 = 25 ms
- Waiting Time P4 = 160 - 35 = 125 ms
- Waiting Time P5 = 130 - 40 = 90 ms

Average Waiting Time (Priority) = $(0 + 60 + 25 + 125 + 90) / 5 = 60$ ms

Round Robin Scheduling

Waiting time

$$P_1 = (0-0) + (25-15) + (45-30) + (70-50) + (90-75) + (110-85)$$
$$= 0 + 10 + 15 + 20 + 15 + 15$$
$$= \boxed{75}$$
$$P_2 = (15-15) + (30-20) + (50-35) + (75-55) + (95-80) + (115-100) + (130-120) + (145-135) + (155-150)$$
$$= 0 + 10 + 15 + 20 + 15 + 15 + 10 + 10 + 5$$
$$= \boxed{100}$$

Waiting time

$$P_3 = (20-15) + (55-25) + (80-60) + (100-85) + (120-105) + (135-125) + (150-140)$$
$$= 5 + 30 + 20 + 15 + 15 + 10 + 10$$
$$= \boxed{105}$$
$$P_4 = (35-35) + (60-40) = 0 + 20 = \boxed{20}$$
$$P_5 = (40-40) + (65-45) + (85-70) + (105-90) + (125-110) + (140-130)$$
$$= 0 + 20 + 15 + 15 + 15 + 10 = \boxed{75}$$

Round Robin Average waiting time = $75 + 100 + 105 + 20 + 75$

$$= 375 / 5$$
$$= \boxed{75}$$

SJF Scheduling has the minimum average waiting time of 36ms.

(Or)

16b	<p>Assume that there are 5 processes, P0 through P4, and 4 types of resources. At T0 we have the following system state:</p> <table><tr><th rowspan="2"></th><th colspan="4">Allocation</th><th colspan="4">Max</th><th colspan="4">Available</th></tr><tr><th>A</th><th>B</th><th>C</th><th>D</th><th>A</th><th>B</th><th>C</th><th>D</th><th>A</th><th>B</th><th>C</th><th>D</th></tr><tr><td>P0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>2</td><td>1</td><td>0</td><td>1</td><td>5</td><td>2</td><td>0</td></tr><tr><td>P1</td><td>1</td><td>2</td><td>3</td><td>1</td><td>1</td><td>6</td><td>5</td><td>2</td><td></td><td></td><td></td><td></td></tr><tr><td>P2</td><td>1</td><td>3</td><td>6</td><td>5</td><td>2</td><td>3</td><td>6</td><td>6</td><td></td><td></td><td></td><td></td></tr><tr><td>P3</td><td>0</td><td>6</td><td>3</td><td>2</td><td>0</td><td>6</td><td>5</td><td>2</td><td></td><td></td><td></td><td></td></tr><tr><td>P4</td><td>0</td><td>0</td><td>1</td><td>4</td><td>0</td><td>6</td><td>5</td><td>6</td><td></td><td></td><td></td><td></td></tr></table> <p>Answer: Need:</p> <table><tr><th></th><th>A</th><th>B</th><th>C</th><th>D</th></tr><tr><td>P0</td><td>0</td><td></td><td></td><td></td></tr><tr><td>P1</td><td>0</td><td></td><td></td><td></td></tr><tr><td>P2</td><td>1</td><td></td><td></td><td></td></tr><tr><td>P3</td><td>0</td><td></td><td></td><td></td></tr><tr><td>P4</td><td>0</td><td></td><td></td><td></td></tr></table> <ul style="list-style-type: none">• Check to see if need0 (0,1,0,0) is less than or equal to work. It is, so let's set finish to true for that process and also update work by adding the allocated resources (0,1,1,0) for that process to work.• Now, let's check to see if need1 (0,4,2,1) <= work. Remember that we have to check each element of the vector need1 against the corresponding element in work. Because 1 is not less than 0 (the fourth element), we need to move on to P2. Need2 (1,0,0,1) is not less than work, so must move on to P3. Need3 (0,0,2,0) is less than work, so we can update work and finish.• Next, let's look at P4. Need4 (0,6,4,2) is less than work, so we can update work and finish• Now we can go back up to P1. Need1 (0,4,2,1) is less than work, so let's update work and finish.• Finally, let's look at P2. Need2 (1,0,0,1) is less than work, so we can then say that the system is in a safe state and the processes will be executed in the following order: Safe Sequence: P0,P3,P4,P1,P2		Allocation				Max				Available				A	B	C	D	A	B	C	D	A	B	C	D	P0	0	1	1	0	0	2	1	0	1	5	2	0	P1	1	2	3	1	1	6	5	2					P2	1	3	6	5	2	3	6	6					P3	0	6	3	2	0	6	5	2					P4	0	0	1	4	0	6	5	6						A	B	C	D	P0	0				P1	0				P2	1				P3	0				P4	0				10	3	3	3	3.2.3
	Allocation				Max				Available																																																																																																																					
	A	B	C	D	A	B	C	D	A	B	C	D																																																																																																																		
P0	0	1	1	0	0	2	1	0	1	5	2	0																																																																																																																		
P1	1	2	3	1	1	6	5	2																																																																																																																						
P2	1	3	6	5	2	3	6	6																																																																																																																						
P3	0	6	3	2	0	6	5	2																																																																																																																						
P4	0	0	1	4	0	6	5	6																																																																																																																						
	A	B	C	D																																																																																																																										
P0	0																																																																																																																													
P1	0																																																																																																																													
P2	1																																																																																																																													
P3	0																																																																																																																													
P4	0																																																																																																																													

Course Outcome (CO) and Bloom's level (BL) Coverage in Questions

