USN | | | | | | | | | | |

Internal Assessment Test 2(Solution and Schemes) – August 2022

Sub:	**Operating System**					Sub Code:	18CS43	Branch:	ISE	
Date:	4/08/2022	Duration:	90 min's	Max Marks:	50	Sem/Sec:	IV A, B & C		OBE	

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1	**Explain the various classical synchronization problems** 1. Reader's writer problem 2. Dining Philosopher problem 3. Bounded Buffer	10	CO1	L2
2	**Explain scheduling algorithms with examples.** 1. First come first serve 2. Shortest Job first (preemptive and non-preemptive) 3. Priority 4. Round Robin Scheduling	10	CO1	L2
3	**Justify the way deadlocks can be prevented by considering four necessary conditions cannot hold.** 1. Mutual exclusion 2. Hold and wait 3. Circular wait 4. No premption	10	CO2	L2

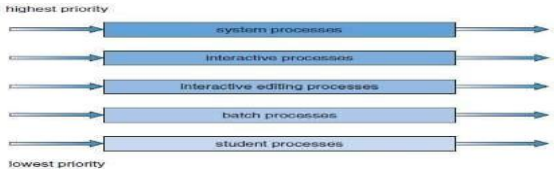
4	**What are semaphores? Explain two primitive semaphore operations. What are its advantages?**
	A semaphore is a synchronization tool is used solve various synchronization problem and can be implementedefficiently.
	Semaphore do not require busywaiting.
	A semaphore S is an integer variable that, is accessed only through two standard atomic operations: wait () and signal (). The wait () operation was originally termed P and signal() was calledV.
	Definition of wait ():
	`wait (S) {`
	`while S <= 0`
	`; // no-op`
	`S--;`
	Definition of signal ():
	`signal (S) {`
	`S++;}`
		10	CO1	L1

5a)	**Explain multilevel and multilevel feedback queue with relevant diagram.** **Multilevel** Useful for situations in which processes are easily classified into different groups. For example, a common division is made between foreground (or interactive) processes and background (or batch) processes.	6	CO1	L2

The ready-queue is partitioned into several separate queues

The processes are permanently assigned to one queue based on some property like memory size process priority or process type. Each queue has its own scheduling algorithm.

For example, separate queues might be used for foreground and background processes.



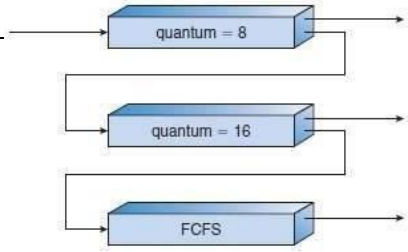
Multilevel feedback queue

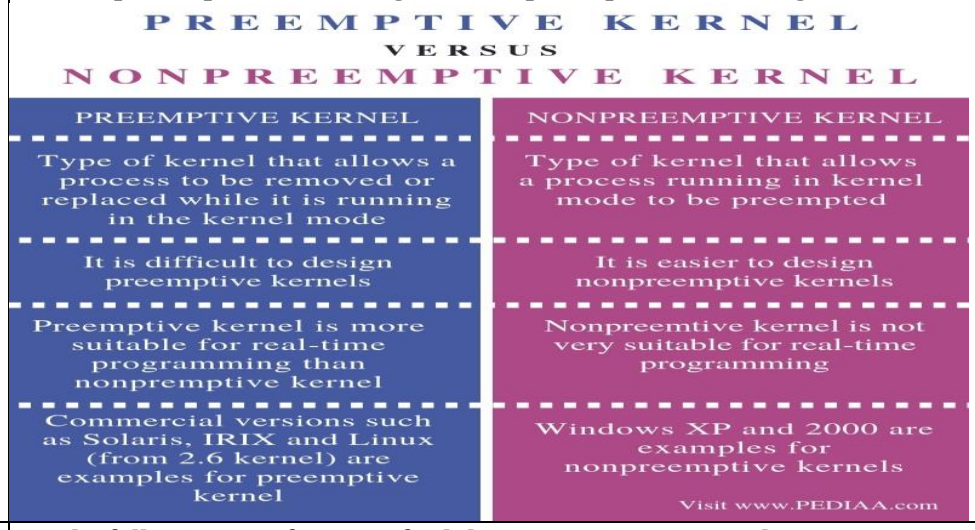
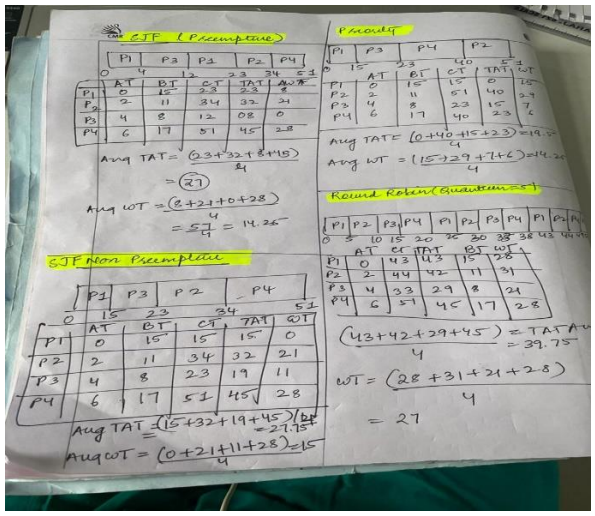
A process may move between queues

The basic idea: Separate processes according to the features of their CPU bursts. For example

If a process uses too much CPU time, it will be moved to a lower-priority queue. This scheme leaves I/O-bound and interactive processes in the higher-priority queues.

If a process waits too long in a lower-priority queue, it may be moved to a higher-priority queue This form of aging prevents starvation.



5b)	**What is pre-emptive scheduling and non-preemptive scheduling?** 	4	CO1	L1
6	**For the following set of process find the avg. waiting time and avg. turn around using Gantt chart for a) Priority b) SJF (preemptive and non- preemptive) c) RR (quantum= 5)** Process | Arrival Time | Burst Time | Priority P1 | 0 | 15 | 1 P2 | 2 | 11 | 4 P3 | 4 | 8 | 2 P4 | 6 | 17 | 3 	10	CO2	L2

Faculty Signature CCI Signature HOD Signature

5b)	What is pre-emptive scheduling and non-preemptive scheduling?	4	CO1	L1
6	For the following set of process find the avg. waiting time and avg. turn around using Gantt chart for a) Priority b) SJF (preemptive and non- preemptive) c) RR (quantum= 5)	10	CO2	L2

Process	Arrival Time	Burst Time	Priority
P1	0	15	1
P2	2	11	4
P3	4	8	2
P4	6	17	3

Faculty Signature CCI Signature HOD Signature