(Affiliated to University of Mumbai)	***End Semester Examination (R-24) FH 2026***
Branch: CE/AIDS/AIML	Course: Operating System
Year/ Semester: SE/VI	Course code: CEC402
Time: 03 hours	Marks: 80

	Note: 1. All questions are compulsory. **2. Figures to right indicate full marks.** **3. Assume suitable data wherever necessary.**	Marks	CO	BL
Q. 1	Attempt any FOUR. (All questions carry equal marks)	**20**		
A.	Explain the Critical Section Problem. Describe the three main requirements that any solution must satisfy. • A **critical section** is a part of a program where shared resources are accessed. • Problem: Multiple processes accessing shared data can cause inconsistency. **Three Requirements:** 1. **Mutual Exclusion:** Only one process can enter critical section at a time. 2. **Progress:** If no process is in CS, decision of next entry should not be delayed. 3. **Bounded Waiting:** A process must not wait indefinitely. **Marking Scheme:** • Definition: 2M • Three Conditions: 3M	05	CO3	L2
B.	Define Thread. Write the differences between user-level and kernel-level threads. • A **thread** is the smallest unit of CPU execution within a process. **Differences:** **Feature** | **User-Level Thread** | **Kernel-Level Thread** Management | User space | OS kernel Speed | Fast | Slower Blocking | Entire process blocks | Only thread blocks Example | Java threads | Windows threads **Marking Scheme:** • Definition: 2M • Differences: 3M	05	CO2	L2
C.	What is thrashing? What is a major cause of it and how to reduce it. • Thrashing is a condition where system spends more time **paging than executing**. **Cause:** • High degree of multiprogramming • Insufficient memory **Reduction:** • Reduce multiprogramming • Use **Working Set Model**	05	CO4	L1

	• Increase RAM **Marking Scheme:** • Definition: 2M • Cause: 1M • Solution: 2M			

D.	Explain the Producer-Consumer Problem using semaphores. Write the pseudo-code for both.
	Semaphores:
	• mutex = 1, empty = n, full = 0
	Producer:
	while(true){
	wait(empty);
	wait(mutex);
	produce_item();
	signal(mutex);
	signal(full);
	}
	Consumer:
	while(true){
	wait(full);
	wait(mutex);
	consume_item();
	signal(mutex);
	signal(empty);
	}
	Marking Scheme:
	• Concept: 2M
	• Pseudo-code: 3M
		05	CO3	L2

E.	Explain the various file access methods in detail. **Types:** 1. **Sequential Access** o Data accessed in order 2. **Direct Access** o Random access using index 3. **Indexed Access** o Uses index block **Marking Scheme:** • Explanation: 3M • Types: 2M	05	CO5	L2
F.	Explain the various Security and Protection Threats in operating systems with suitable examples. **Types:** • **Virus:** Attaches to programs • **Worm:** Self-replicating • **Trojan Horse:** Hidden malicious code • **Denial of Service (DoS):** Overloads system	05	CO6	L2

	• **Unauthorized Access** **Marking Scheme:** • Explanation: 3M • Examples: 2M			
Q.2	Attempt any FOUR. (All questions carry equal marks)	**40**		
A.	Consider the following reference string: 1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6. Find the number of page faults with FIFO, Optimal Page Replacement and LRU with frame size=4. Illustrate which technique is best for given example. **FIFO Page Faults = 14** **LRU Page Faults = 12** **Optimal Page Faults = 9** **Conclusion:** • **Best Algorithm = Optimal (minimum faults)** **Marking Scheme:** • FIFO: 3M • LRU: 3M • Optimal: 3M • Conclusion: 1M	10	CO4	L3
B.	Consider a system that contains five processes P1, P2, P3, P4, P5 and the three resource types A, B and C. Following are the resources types: A has 10, B has 5 and the resource type C has 7 instances. Calculate available, need matrix and Whether system will be in safe state or not by using Banker's Algorithm. **Need Matrix = Max – Allocation** **Process Need** P1 (7,4,3) P2 (1,2,2) P3 (6,0,0)	10	CO3	L3

Process	**Allocation**			**Max**		
	A	**B**	**C**	**A**	**B**	**C**
P1	0	1	0	7	5	3
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

	P4 (0,1,1) P5 (4,3,1)			
	Safe Sequence: → P2 → P4 → P5 → P1 → P3 **✓ System is SAFE** **Marking Scheme:** • Available: 2M • Need: 3M • Safe Sequence: 3M • Conclusion: 2M			
C.	Explain Protection Structures in detail. Describe Granularity of Protection, Access Control Matrix, and Protection Domain with neat diagrams and examples. **1. Granularity** • Level of access (file, record, byte) **2. Access Control Matrix** **Subject File1 File2** User1 R/W R User2 R W **3. Protection Domain** • Set of access rights for a process **Marking Scheme:** • Concepts: 6M • Diagram: 4M	10	CO6	L2
D.	Explain System call. What are different types of it (with example). **Definition:** • Interface between user and OS **Types:** • Process Control (fork, exit) • File Management (open, read) • Device Management • Information Maintenance • Communication **Marking Scheme:** • Definition: 2M • Types: 8M	10	CO1	L2
E.	What are the different memory management techniques? • Paging • Segmentation • Swapping • Virtual Memory **Marking Scheme:**	10	CO4	L2

	• Explanation: 6M • Types: 4M			
F.	Distinguish between preemptive and non-preemptive scheduling. Explain each type with an example. **Type Preemptive Non-Preemptive** CPU Control Can interrupt Cannot interrupt Example Round Robin FCFS **Different Conditions** **Marking Scheme:** • Difference: 5M • Example: 5M	10	CO2	L3
Q.3	Attempt any FOUR Short note on.	**20**		
A.	Explain the Evolution of Operating Systems from first generation to modern systems. Highlight the major advancements in each generation with suitable examples. **Solution:** • **First generation (1940s–1950s):** No real operating system existed; computers were programmed manually using machine language, plugboards, and punched cards. The main goal was to make the machine work, but CPU time was heavily wasted. • **Second generation (1950s–1960s):** Batch processing systems were introduced, so similar jobs were grouped and executed one after another without user interaction. This improved CPU utilization; examples include early batch systems and IBM OS/360. • **Third generation (1960s–1970s):** Multiprogramming and time-sharing systems appeared, allowing multiple programs and users to share the CPU. This made computers interactive and more efficient; examples include CTSS, Multics, and UNIX. • **Fourth generation (1980s–1990s):** Personal computers became common, and operating systems focused on user-friendliness, GUIs, and multitasking. Examples include MS-DOS, Windows, and Macintosh OS. • **Modern systems (2000s to present):** Operating systems now support networking, virtualization, mobile computing, cloud integration, security, and multi-core processing. Examples include Linux, Windows 11, Android, and iOS. • **Major advancement overall:** OS evolution moved from **manual control** to **batch processing**, then **interactive multitasking**, then **graphical and networked systems**, and finally to **smart, mobile, cloud-connected platforms**.	05	CO1	L2

B.	The five-state process model New, Ready, Running, Waiting, Terminated Dig-2m Explain-3m	05	CO2	L2
C.	Deadlock and its conditions **Conditions:** • Mutual Exclusion • Hold & Wait • No Preemption • Circular Wait Def-1m Condition-4m	05	CO3	L2
D.	IPC Shared Memory Message Passing Def-1m Types-4m	05	CO2	L2
E.	File Allocation Methods Contiguous Linked Indexed Types-1m explain-4m	05	CO5	L2
F.	Types of Process schedulers Types-1m explain-4m	05	CO2	L2
********* **All the Best********** 				