Reg. No: 24BCI0230

VIT
Vellore Institute of Technology

Final Assessment Test – November 2026

Course: **BCSE303L - Operating Systems**

Class NBR(s): **2786, 2793, 2798, 2801, 2805, 2812, 2830, 2838, 2841, 2843, 2846, 2849, 2857, 2864, 2869, 2873, 2875, 2878, 2880, 2882, 2885, 2891, 2999, 3034**

Slot: **B2+TB2**

Time: **3 Hours**

Max. Marks: **100**

➢ KEEPING MOBILE PHONE/ANY ELECTRONIC GADGETS, EVEN IN 'OFF' POSITION IS TREATED AS EXAM MALPRACTICE

➢ DON'T WRITE ANYTHING ON THE QUESTION PAPER

General Instruction

Course Outcomes:

CO1: Interpret the evolution of OS functionality, structures, layers and apply various types of system calls of various process states

CO2: Design scheduling algorithms to compute and compare various scheduling criteria

CO3: Apply and analyze communication between inter process and synchronization techniques.

CO4: Implement page replacement algorithms, memory management problems and segmentation

CO5: Differentiate the file systems for applying different allocation, access technique, representing virtualization and providing protection and security in OS

BL – Blooms Taxonomy Level (1 – Remember, 2 – Understand, 3 – Apply, 4 – Analyse, 5 – Evaluate, 6 – Create)

PART-A (10 X 10 = 100 Marks)

Answer all Question(s)

Q1. (a) Classify each of the following instructions as privileged (kernel mode only) or non-privileged (allowed in user mode). Justify each in one line. **[6 Marks]** — **10M** **CO1** **BL1**

1. Read the current system date and time
2. Modify the base and limit registers of the memory-protection hardware
3. Execute a trap (system-call) instruction
4. Halt the CPU
5. Directly write to an I/O device port
6. Load a value into a general-purpose register

(b) Explain how dual-mode operation is implemented using a mode bit. What happens when a user program attempts to execute a privileged instruction? **[4 Marks]**

Q2. **(a)** Explain the following operating system structures with an example in each category: monolithic, layered, and microkernel. **[6 marks]** — **10M** **CO2** **BL3**

(b) Compare the monolithic and microkernel approaches on performance, reliability, security and extensibility. **[4 marks]**

Q3. A system has four processes (P1 to P4) and four resource types (R1 to R4). The current allocation and maximum needs are:

10M CO4 BL3

Process	Allocation (R1 R2 R3 R4)	Maximum (R1 R2 R3 R4)
P1	2011	3122
P2	1100	2210
P3	0111	1231
P4	1010	1121

Available = (1, 1, x, 1).

(a) Compute the Need matrix. **[2 marks]**

(b) Find the smallest value of **x** for which the system is in a safe state, and give a safe sequence. **[6 marks]**

(c) With x equal to that value, P3 requests (1, 0, 1, 0). Can the request be granted immediately? Justify. **[2 marks]**

Q4. A machine has 40-bit virtual addresses and 32-bit physical addresses. The page size is 8 KB, and each page table entry (PTE) is 4 bytes. **10M CO4 BL4**

(a) If a single-level page table is used, how many entries does it have, and what is its size per process? Explain why this is impractical. **[3 marks]**

(b) The designer decides to use a multi-level page table in which every page table fits exactly in one page. Determine the number of levels and show how the virtual address is divided. **[2 marks]**

(c) The system has a TLB with 16 entries. A program's instructions fit in one page, and it reads 4-byte integers sequentially from an array that spans thousands of pages. Ignore instruction fetches and consider data references only.

- Estimate the TLB hit ratio and explain why the TLB is effective or ineffective in this case.
- Using the page table from (b), find the effective memory access time (EMAT). TLB lookup takes 10 ns, and one memory access takes 100 ns. **[3 marks]**

(d) Define *TLB reach (total amount of memory that a CPU's Translation Lookaside Buffer can map at one time without requiring a slow page table lookup)* and compute it for this TLB. Explain why the same program would perform poorly if it accessed array elements at random positions across the whole array. **[2 marks]**

Q5. (a) Explain demand paging. Describe, step by step, how the operating system handles a page fault. **[5 marks]** **10M CO5 BL2**

(b) Define thrashing. Explain how it arises and why it reduces CPU utilization. Explain the working-set model. How does it help to prevent thrashing? **[5 marks]**

Q6. (a) Explain why classic x86 processors could not be virtualized by simple trap-and-emulate, and describe how binary translation solves this in full virtualization. Include the role of the translation cache. **[4 marks]**

(b) A hypervisor translates guest kernel code one basic block at a time and then executes it. Could it translate the entire program before running it? Give two technical reasons that make this difficult, and compare the two

techniques by giving two advantages and two disadvantages of each. Compare full virtualization and paravirtualization. **[6 marks]** **10M** **CO5** **BL4**

Q7. (a) Describe how a file-system directory can be organized as (i) a linear list and (ii) a hash table. For each method, mention one benefit and one drawback. **[4 marks]** **10M** **CO5** **BL3**

(b) Explain contiguous, linked, and indexed allocation methods for disk space. For each method, describe how a file is stored and discuss its advantages and disadvantages. **[6 marks]**

Q8. (a) Describe the physical structure of a magnetic disk (platter, track, sector, cylinder) and how logical block addressing works. Name the possible ways a disk can be attached to a computer. **[4 marks]** **10M** **CO5** **BL3**

(b) A disk has 200 cylinders (0 to 199). The head is at cylinder 100 and moving towards higher cylinder numbers. The pending request queue is: 23, 89, 132, 42, 187, 130, 64, 120. Find the total head movement for FCFS, SSTF, SCAN and C-SCAN, and state which gives the least movement. **[6 marks]**

Q9. **a)** Five philosophers sit around a table with one chopstick *(semaphore, initialized to 1)* between each pair of neighbors. Philosopher i uses *chopstick[i]* and *chopstick[(i+1) % 5]*.

Attempt 1. To avoid hold-and-wait, a student rewrites the entry section so that a philosopher gives up its first chopstick if the second is not free:

```
while (true) {
    wait(chopstick[i]);
      if (trywait(chopstick[(i+1) % 5]) == FAILED) {    /*
non-blocking */
        signal(chopstick[i]);                            /* put
it back */
         continue;                                        /*
retry */
    }
    /* eat */
    signal(chopstick[i]);
    signal(chopstick[(i+1) % 5]);
    /* think */
}
```

(a) Is deadlock possible in Attempt 1? Show an execution in which the philosophers still make no progress, and name this phenomenon. Explain how it differs from deadlock. **[3 marks]**

Attempt 2. The student now numbers the chopsticks 0 to 4 and uses this entry rule: every philosopher picks up the lower-numbered chopstick first. Philosophers 0 to 3 therefore behave as in the original code, but philosopher 4 must pick up *chopstick[0]* before *chopstick[4]*.

(b) Prove that Attempt 2 is deadlock-free. Identify which of the four deadlock conditions is eliminated, and explain why this method does not require the philosophers to know the other philosophers' states. **[3 marks]**

(c) The following monitor solution (states THINKING, HUNGRY, EATING, with *test()* allowing a philosopher to eat only if neither neighbor is eating) is deadlock-free, but starvation is possible. **[4 marks]**

- Create a schedule in which philosopher 2 is starved by philosophers 1 and 3. Show the state of each relevant philosopher at each step.
- Modify *pickup()* and *test()* so that starvation is prevented, and justify why your solution works.

10M **CO3** **BL**

(OR)

b) Peterson's solution for two processes, P0 and P1, uses the shared variables *int turn; boolean interested[2];*. Its entry section for process *i (with j = 1 - i)* is: **10M** **CO3** **BL**

```
interested[i] = TRUE;
turn = i;
while (turn == i && interested[j] == TRUE);
```

and its exit section is *interested[i] = FALSE;*

(a) Explain how this solution satisfies mutual exclusion, progress, and bounded waiting. **[5 marks]**
(b) A system uses a non-preemptive scheduler. Give a scenario in which Peterson's solution fails, and state the exact reason. Would the same scenario be a problem under preemptive scheduling? **[3 marks]**
(c) A programmer swaps the first two lines of the entry section, so that *turn = i;* comes before *interested[i] = TRUE.* Show an interleaving in which mutual exclusion is violated. **[2 marks]**

Q10. a) (a) Define a distributed file system (DFS). Explain naming transparency, location transparency and location independence. **[5 marks]** **10M** **CO5** **BL2**
(b) Explain client-side caching in a DFS. Discuss cache location, cache-update policies and consistency approaches. **[5 marks]**

(OR)

b) (a) Explain the concept of a protection domain. Describe the access-matrix model with an example. **[3 marks]** **10M** **CO5** **BL2**
(b) Explain the four ways of implementing an access matrix (global table, access lists, capability lists, and lock-key). Compare access lists with capability lists. **[4 marks]**
(c) Consider the access matrix below. **[3 marks]**

Domain	F1	F2	Printer
D1	read	-	-
D2	-	read, write	print
D3	read, write	read	-

(i) Write the access list of F2 and the capability list of D3.
(ii) Describe two schemes for revoking a capability in a capability-based system.

⇔⇔⇔⇔⇔⇔