REG.NO.:

VIT
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science Engineering and Information Systems
CONTINUOUS ASSESSMENT TEST - II
FALL SEMESTER 2024-2025

SLOT: G1+TG1

**Programme Name & Branch** : M. Tech
**Course Code and Course Name** : ISWE204L Operating Systems
**Faculty Name(s)** : Dr. PRAVEEN KUMAR REDDY M, Dr. HARSHITA PATEL, Dr. ARIVUSELVAN K, Dr. SIVASHANKARI R, Dr. KRISHNAMOORTHY N
**Class Number(s)** : VL2024250102906, VL2024250103719, VL2024250102887, VL2024250102900, VL2024250102926
**Date of Examination** :
**Exam Duration** : 90 minutes **Maximum Marks: 50**

Answer All Questions

| Q. No | Question | M | CO | BL |
|---|---|---|---|---|
| 1. | Assume that there are 5 processes, $P_0$ through $P_4$, and 4 types of resources. At $T_0$ we have the following system state:<br>Max Instances of Resource Type A = 3<br>Max Instances of Resource Type B = 17<br>Max Instances of Resource Type C = 16<br>Max Instances of Resource Type D = 12<br>(Given Matrices below)<br>Create the need matrix (max-allocation) and use the safety algorithm to test if the system is in a safe state or not? Give the Safe Sequence Order? | 10 | 2 | 2 |

**Given Matrices**

| | Allocation Matrix (N0 of the allocated resources By a process) | | | | Max Matrix Max resources that may be used by a process | | | | Available Matrix Not Allocated Resources | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | A | B | C | D | A | B | C | D | A | B | C | D |
| $P_0$ | 0 | 1 | 1 | 0 | 0 | [illegible] | 1 | [illegible] | 1 | 5 | 2 | [illegible] |
| $P_1$ | 1 | 2 | 3 | 1 | 1 | 6 | 5 | [illegible] | | | | |
| $P_2$ | 1 | 3 | 6 | 5 | 2 | 3 | 6 | [illegible] | | | | |
| $P_3$ | 0 | 6 | 3 | 2 | 0 | 6 | 5 | [illegible] | | | | |
| $P_4$ | 0 | 0 | 1 | 4 | 0 | 6 | 5 | 6 | | | | |
| Total | 2 | 12 | 14 | 12 | | | | | | | | |

| Q. No | Question | M | CO | BL |
|---|---|---|---|---|
| 2. | a) Consider the set of 5 processes whose arrival time and burst time are given below. If the CPU scheduling policy is Round Robin with time quantum = 2 unit, calculate the average waiting time and average turnaround time. | 10 | 3 | 3 |

| Process | Arrival time | Burst Time |
|---|---|---|
| P1 | 0 | 5 |
| P2 | 1 | 3 |
| P3 | 2 | 1 |
| P4 | 3 | 2 |
| P5 | 4 | 4 |

b) Consider the set of 5 processes whose arrival time and burst time are given below. If the CPU scheduling policy is Round Robin with time quantum = 3 unit, calculate the average waiting time and average turnaround time.

| Process | Arrival time | Burst Time |
|---|---|---|
| P1 | 1 | 4 |
| P2 | 3 | 5 |
| P3 | 4 | 3 |
| P4 | 0 | 2 |
| P5 | 6 | 3 |

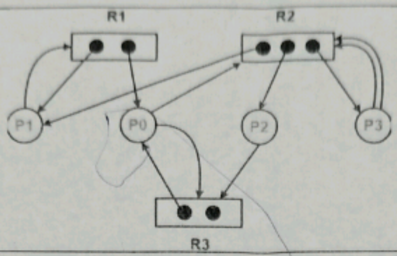
| Q. No | Question | M | CO | BL |
|---|---|---|---|---|
| 3. | a) Consider the resource allocation graph in the figure and find if the system is in a deadlock state or not | 5 | 2 | 4 |

REG.NO.:

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science Engineering and Information Systems
CONTINUOUS ASSESSMENT TEST - II
FALL SEMESTER 2024-2025

SLOT: G1+TG1



b) Consider a system with m resources of the same type, shared by three processes X, Y, and Z, which have peak demands of 5, 7, and 9 respectively. For what value of m will deadlock not occur? — 5

4. Fetch_And_Add(X,i) is an atomic Read-Modify-Write instruction that reads the value of memory location X, increments it by the value i and, returns the old value of X. It is used in the pseudo code shown below to implement a busy-wait lock. L is an unsigned integer shared variable initialized to 0. The value of 0 corresponds to lock being available, while any non-zero value corresponds to the lock being not available. — 10 | 2 | 4

```
AcquireLock(L)
{
  while (Fetch_And_Add(L,1))
  L = 1;
}
ReleaseLock(L)
{
  L = 0;
}
```

Check for mutual exclusion by analyzing if multiple processes can enter their critical sections simultaneously. Verify the progress property by checking if processes can eventually enter their critical sections when they want to. Provide detailed justification for both properties

5. Consider the following implementation for process synchronization: — 10 | 2 | 5

```
// Shared variables
boolean flag[2] = {false, false}   // Initially both false
int turn;                  // Shared variable indicating which process's turn
// Code for Process i (where i is 0 or 1)
do {
   flag[i] = true;
   turn = j;
   while (turn == j);
      // CRITICAL SECTION
   flag[i] = false;
      // REMAINDER SECTION
 } while (true);
```

Analyze the implementation and provide a detailed justification of why mutual exclusion is maintained or violated. Justify your conclusion with a detailed explanation of how the algorithm ensures (or fails to ensure) progress. Analyze the possibility of deadlock or starvation