# CS330-001 Introduction to Operating System
## Assignment 3

1. (5 marks) The producer and consumer processes share the following data structures:
int n;
semaphore mutex = 1;
semaphore empty = n;
semaphore full = 0
The code for the producer process and the consumer process is shown below. Describe what changes would be
necessary to the code so that a mutex lock could be used instead of a binary semaphore.

```
-------------------------------------------------------------------------------------------
while (true) { // The structure of Producer Process
. . .
/* produce an item in next_produced */
. . .
wait(empty);
wait(mutex);
. . .
/* add next_produced to the buffer */
. . .
signal(mutex);
signal(full);
}
-------------------------------------------------------------------------------------------
while (true) { // The structure of Consumer Process
wait(full);
wait(mutex);
. . .
/* remove an item from buffer to next_consumed */
. . .
signal(mutex);
signal(empty);
. . .
/* consume the item in next_consumed */
. . .
}
-------------------------------------------------------------------------------------------
```

2. (10 marks) CPU Scheduling

   Consider the following set of processes, with the length of the CPU burst given in milliseconds:

| Process | Burst Time | Priority |
|---|---|---|
| *P1* | 5 | 4 |
| *P2* | 3 | 1 |
| *P3* | 1 | 2 |
| *P4* | 7 | 2 |
| *P5* | 4 | 3 |

   The processes are assumed to have arrived in the order *P*1, *P*2, *P*3, *P*4, *P*5, all at time 0.

   a) Draw four Gantt charts that illustrate the execution of these processes using the following scheduling algorithms: FCFS, SJF, non preemptive priority (a larger priority number implies a higher priority), and RR (quantum = 2).

   b) What is the turnaround time of each process for each of the scheduling algorithms in part a?

   c) What is the waiting time of each process for each of these scheduling algorithms?

   d) Which of the algorithms results in the minimum average waiting time (over all processes)?

3. (10 Marks) CPU Scheduling
   The following processes are being scheduled using a preemptive, round robin scheduling algorithm.

| Thread | Burst Time | Priority | Arrival |
|---|---|---|---|
| *P1* | 8 | 15 | 0 |
| *P2* | 3 | 20 | 5 |
| *P3* | 10 | 15 | 20 |
| *P4* | 4 | 20 | 22 |
| *P5* | 5 | 18 | 25 |
| *P6* | 6 | 15 | 30 |

Each process is assigned a numerical priority, with a higher number indicating a higher relative priority. The scheduler will execute the highest-priority process. For processes with the same priority, a round-robin scheduler will be used with a time quantum of 4 units. If a process is preempted by a higher priority process, the preempted process is placed at the end of the ques.

a) Show the scheduling order of the processes using a Gantt chart.

b) What is the turnaround time for each process?

c) What is the waiting time for each process?

4. (10 marks) Consider the following snapshot of a system:

| | ***Allocation*** | ***Max*** | ***Available*** |
|---|---|---|---|
| | *A B C D* | *A B C D* | *A B C D* |
| *T0* | 0 0 1 2 | 0 0 1 2 | 1 5 2 0 |
| *T1* | 1 0 0 0 | 1 7 5 0 | |
| *T2* | 1 3 5 4 | 2 3 5 6 | |
| *T3* | 0 6 3 2 | 0 6 5 2 | |
| *T 4* | 0 0 1 4 | 0 6 5 6 | |

Answer the following questions using the banker's algorithm:

a) What is the content of the matrix Need?

b) Is the system in a safe state?

c) If a request from thread T1 arrives for (0,4,2,0), can the request be granted immediately?

5. (10 marks) Consider the following snapshot of a system:

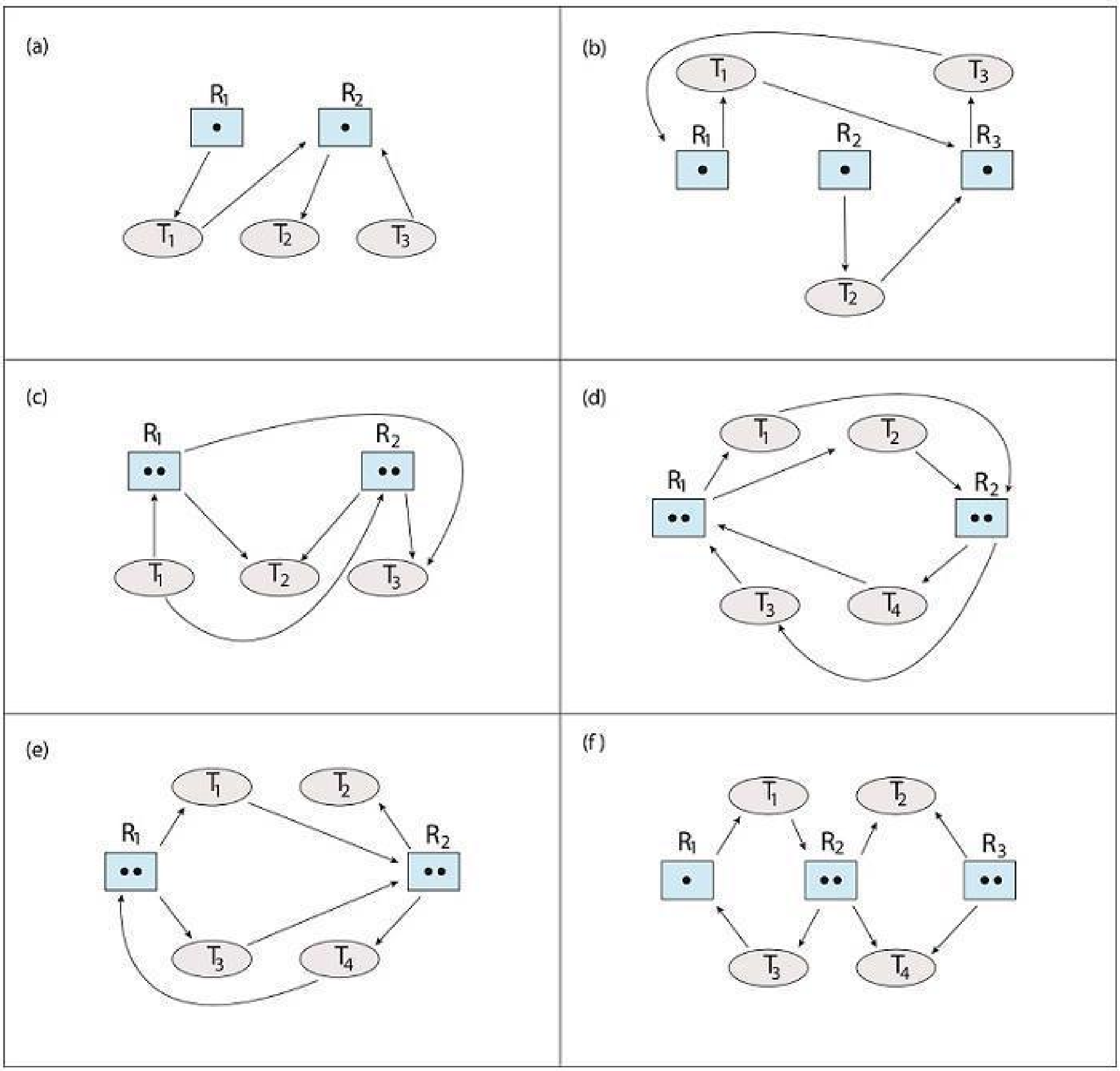
| | ***Allocation*** | ***Max*** |
|---|---|---|
| | *A B C D* | *A B C D* |
| *T 0* | 3 0 1 4 | 5 1 1 7 |
| *T1* | 3 2 1 0 | 3 2 1 1 |
| *T2* | 3 1 2 1 | 3 2 2 1 |
| *T3* | 0 5 1 0 | 4 6 1 2 |
| *T4* | 4 2 1 2 | 6 3 2 5 |

Using the banker's algorithm, determine whether or not each of the following states is unsafe. If the state is safe, illustrate the order in which the threads may complete. Otherwise, illustrate why the state is unsafe.

a) Available = (0, 3, 0, 1)

b) Available = (1, 0, 0, 2)

6. (15 marks) Which of the six resource-allocation graphs shown in the figure illustrate the deadlock? For those situations that are deadlocked, provide the cycle of threads and resources. Where there is not a deadlock situation, illustrate the order in which the threads may complete execution.




7. (10 marks) Dining-philosophers Problem

a) Describe how deadlock is possible with the dining-philosophers problem.

b) Consider the version of the dining-philosophers problem in which the chopsticks are placed at the center of the table and any two of them can be used by a philosopher. Assume that requests for chopsticks are made one at a time. Describe a simple rule for determining whether a particular request can be satisfied without causing deadlock given the current allocation of chopsticks to philosophers.