National University of Singapore
School of Computing

CS2106 **Tutorial 6** Semester 1 11/12

1. The four conditions for deadlock: mutual exclusion, hold and wait, no preemption, and circular waiting are both sufficient and necessary for deadlock if there are one copy of each resource type.

   If there can be more than one copy of each resource type, however, the four conditions are necessary but not sufficient for deadlock. In other words, it is possible that the four conditions hold, but there is no deadlock.

   Construct an example using three processes A, B, C, and two resource types R and S (with one copy of R and two copies of S) to show that there can be no deadlock even if the four conditions for deadlock hold.

2. Consider the case where each resource type has only one copy. Is the following statement correct?

   "Only processes that are part of a cycle in a resource allocation graph are in a deadlock."

3. Consider the following snapshot of a system's states, with three processes and five allocatable resources. The current allocated and requested resources are as follows:

		Available
		0 0 x 1 1
	Allocated	Requested
Process A	2 0 1 1 0	0 2 1 0 0
Process B	1 1 0 1 0	1 0 3 0 0
Process C	1 1 1 1 0	0 0 1 1 1

   What is the minimum value of x in order for the processes to be deadlock-free?

4. A non-deadlock solution to the dining philosopher problem is to allow preemption of chopsticks. If a more senior philosopher P_s is waiting for a chopstick that is being used by a junior philosopher P_j, P_s can pull rank on P_j and preempt the chopstick from P_j. P_j has to pass the chopstick to P_s so that P_s can eat while P_j waits.

   This scheme is obviously unfair to a junior philosopher if he happens to sit between two senior philosophers, as he gets to eat less often than his senior colleagues.

   Suggest a improvement to the above preemption-based solution to the dining philosopher problem to improve the fairness among the philosophers.

5. ($\star$) What are the necessary and sufficient conditions for deadlock if there can be more than one copy of each resource type? (Hint: Extend the circular waiting condition).