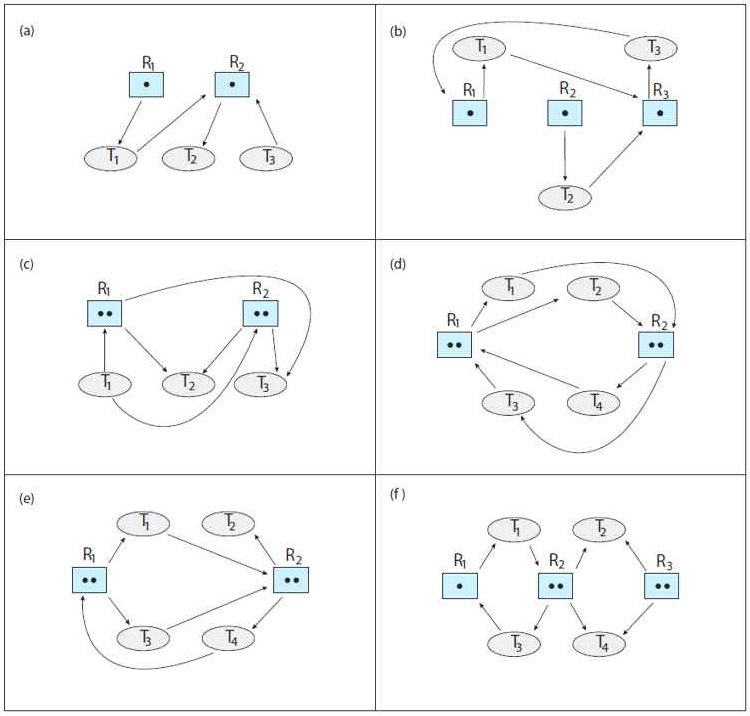
1. 다음 여섯개의 resource-allocation graph의 그림 중에서 데드락이 발생한 경우만 모아 놓은 것을 고르시오.



1) a, b, d
2) b, c, d
3) b, d
4) b, d, e, f
5) b, c, d, e, f

2. 어떤 시스템이 세 개의 쓰레드 T1, T2, T3로 구성되어 있고 이 쓰레드들은 동일한 자원인 R의 인스턴스 세 개를 필요로 하고 있다. 이 시스템이 절대로 데드락에 빠지지 않게 하기 위해서는 R이 최소한 몇 개 이상의 인스턴스를 가져야 하는가?

1) 6
2) 7
3) 8
4) 9
5) 10

3. 데드락 방지 (deadlock prevetion) 방법에 대한 설명으로 가장 틀린 설명은?

   1) 새로운 자원을 요청(request)하기 전에 모든 자원을 반납(release)하면 데드락은 발생하지 않는다.

   2) 모든 자원에 유일한 번호를 부여하고, 자원을 요청할 때 반드시 오름차순으로 요청을 하면 데드락을 발생하지 않도록 할 수 있다.

   3) 새로운 자원을 요청하는 쓰레드가 있으면 해당 자원을 소유한 쓰레드를 선점(preemption)시켜 버리는 데드락이 발생하지 않는다.

   4) 뮤텍스락이나 세마포어를 이용하여 상호 배제(mutual exclusion)를 하도록 하면 데드락은 발생하지 않는다.

4. 한 집합의 쓰레드들이 공유하는 자원이 단 하나의 인스턴스를 가진 자원 딱 하나일 경우에 대한 설명으로 가장 옳은 것은?

   1) 데드락이 발생할 일이 전혀 없다.

   2) 두 개 이상의 쓰레드가 이 자원에 대한 경쟁 상황이 발생하면 데드락이 발생한다.

   3) 세 개 이상의 쓰레드가 이 자원에 대한 경쟁 상황이 발생하면 데드락이 발생한다.

   4) 한 쓰레드가 이 자원을 점유하고 있을 때 다른 쓰레드가 이 자원을 요청하는 상황이 되면 데드락이 발생한다.

5. 다음 중 데드락이 발생할 수 있는 조건 네 가지와 거리가 가장 먼 것은?

   1) 상호 배제 (mutual exclustion)

   2) 점유 대기 (hold and wait)

   3) 선점 불가 (no preemption)

   4) 한정 대기 (bounded waiting)

6. 교착상태(deadlock)의 방지(prevention)와 회피(avoidance)와 관련하여 가장 적절한 설명은?

   1) 데드락 회피(avoidance)를 위해서, 어떤 요청을 수락(grant)했을 때의 시스템 상태가 safe state라면 그 요청을 수락해도 된다.

   2) 데드락 방지(prevention)을 위해서, 어떤 시스템의 상태가 safe state에 있을 때 자원 요청이 들어오면 그 요청을 수락해도 된다.

   3) 데드락 방지(prevention)는 데드락을 발생하지 않도록 하고, 데드락 회피(avoidance)는 데드락이 발생하도록 두고, 데드락이 발생하면 이를 감지하여 복구한다.

   4) 데드락 회피(avoidance)를 위한 알고리즘은 어떤 시스템의 요청(request)에 대한 선행지식(piori knowledge)을 필요로 하지 않는다.

7. 다음 시스템 스냅샷을 고려해보자.

   쓰레드 집합 T = { T0, T1, T2 }, 자원 집합 R = { A, B, C }
   세 개의 자원 유형 A, B, C의 인스턴스 개수는 각각 (8, 5, 4)이다.

   Allocation은 현재 프로세스가 점유한 자원의 수이고,
   Max는 각 프로세스가 필요로 하는 자원 수의 최대값이라고 할 때,
   Banker's Algorithm을 적용했을 때 올바른 설명으로만 묶인 것은?

	Allocation			Max		
	A	B	C	A	B	C
T0	0	0	1	8	4	3
T1	3	2	0	6	2	0
T2	2	1	1	3	3	3

   1) Banker's Algorithm에서 Available의 값은 (3, 2, 3)이다.
   2) T0 쓰레드의 Need 벡터 (Need[0]) 의 값은 (8, 4, 3)이다.
   3) 현재 이 시스템의 상태는 safe state이다.
   4) 위 상태에서 T0가 자원을 다음과 같이 요청했다.(A=2, B=0, C=2)
      Request[0] = (2, 0, 2)
      이 때 Resource-Request 알고리즘을 적용하면 Allocation[0] 벡터의 값은 (1, 0, 3)이 된다.
   5) 위 상태에서 T1이 자원을 다음과 같이 요청했다. (A=2, B=0, C=0)
      Request[1] = (2, 0, 0)
      이 때 Safety 알고리즘을 적용하면 Work 벡터의 값은 (8, 5, 4)가 된다.

8. 만약 어떤 시스템이 4개의 인스턴스를 가진 자원을 하나 가지고 있다고 해보자. 이 때 3개의 쓰레드가 이 자원을 공유하고 있는데, 이 쓰레드는 2개의 인스턴스가 필요한 상황이다. 이런 시스템 상황에 대한 설명으로 가장 옳은 것은?

   1) 한 쓰레드가 2개를 점유하고 나면, 2개의 자원만 남기 때문에 데드락이 발생한다.
   2) 한 쓰레드가 한 개씩의 자원을 점유하고 나면, 1개의 자원만 남기 때문에 데드락이 발생한다.
   3) 두 개의 쓰레드가 두 개씩의 자원을 점유하고 나면, 한 개의 쓰레드가 자원을 얻지 못해 데드락이 발생한다.
   4) 이런 시스템의 상황에서는 데드락이 발생할 수가 없다.

9. 다음은 연습문제 8.3의 시스템 스냅샷이다.

	Allocation	*Max*	*Available*
	A B C D	*A B C D*	*A B C D*
P_0	0 0 1 2	0 0 1 2	1 5 2 0
P_1	1 0 0 0	1 7 5 0	
P_2	1 3 5 4	2 3 5 6	
P_3	0 6 3 2	0 6 5 2	
P_4	0 0 1 4	0 6 5 6	

위 시스템 상황에 대해 올바른 설명으로만 묶여진 것은?

a) P4의 Need 벡터는 Need[4] = (0, 6, 4, 2)이다.
b) 이 시스템의 상태는 safe state이다.
c) Request1 = (0, 4, 2, 0) (P1의 자원요청)에 대해서 grant해도 된다.
d) 위 c)번의 Request1의 요청을 처리한 후에 Available1은 (1, 1, 0, 0)이 된다.
e) 프로세스 시퀀스 〈P0, P2, P3, P1, P4〉의 순서는 safe sequence이다.

1) a, b, c, d, e
2) a, b, d, c
3) a, b, e
4) a, b, c, e
5) a, b, d, e

Answers (indended by the Question Provider):

1) 3
2) 2
3) 4
4) 1
5) 4
6) 1
7) 3
8) 4
9) 1