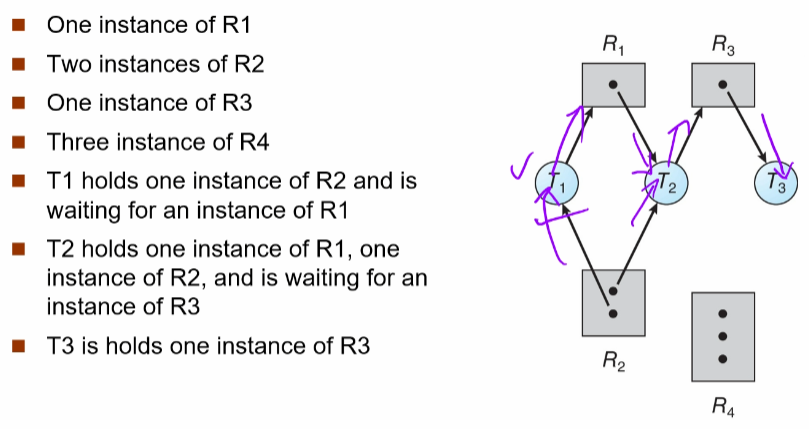
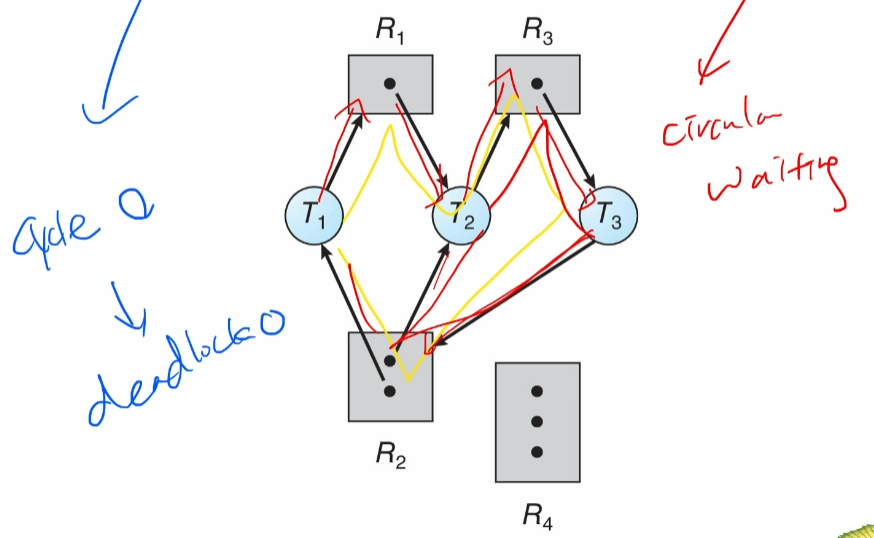
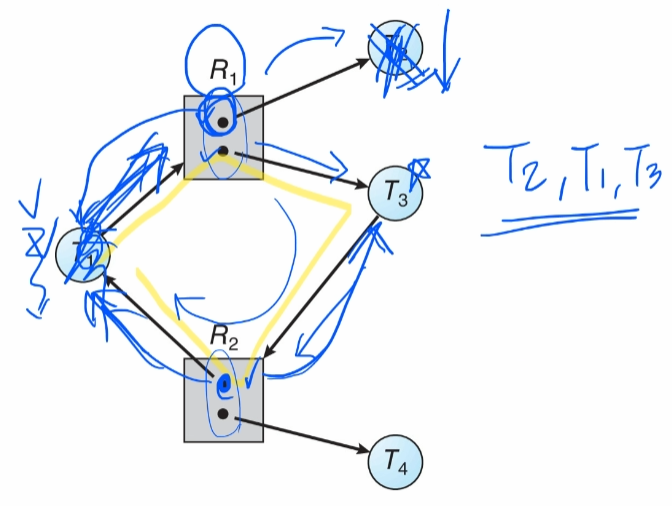
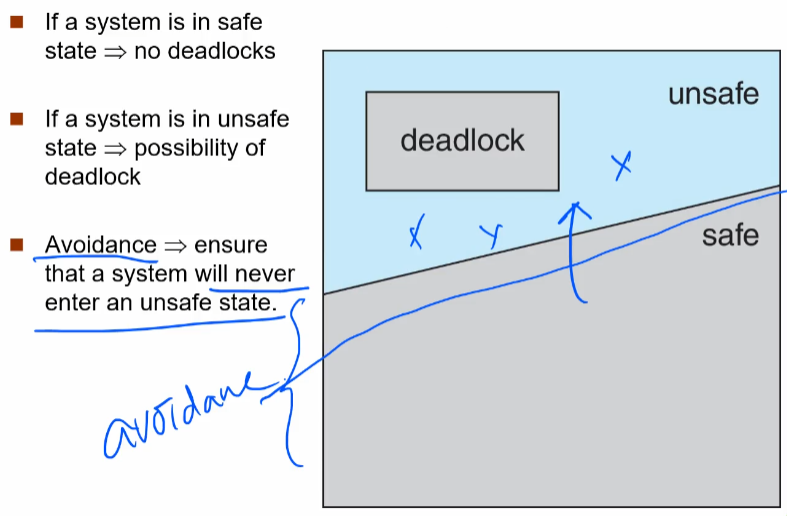
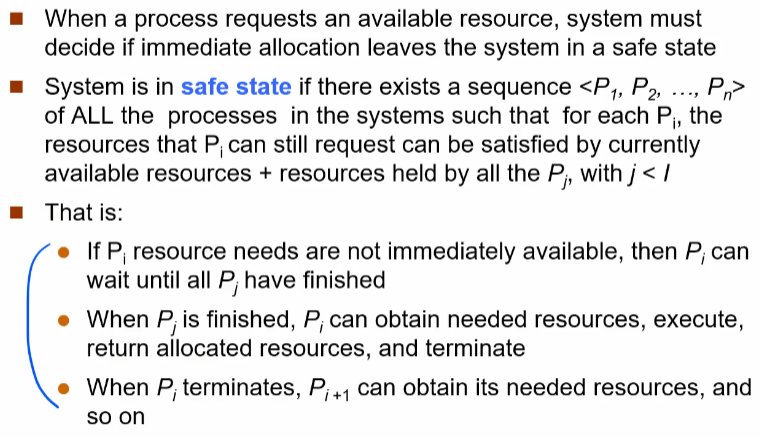
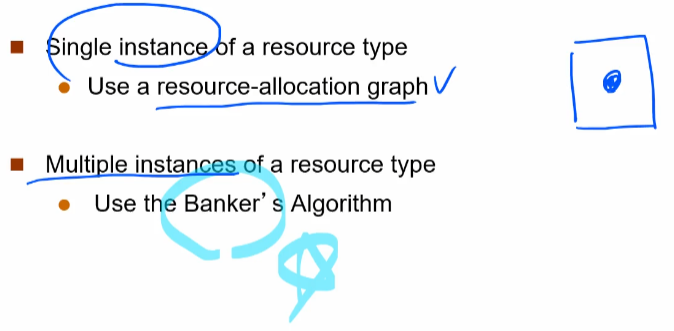
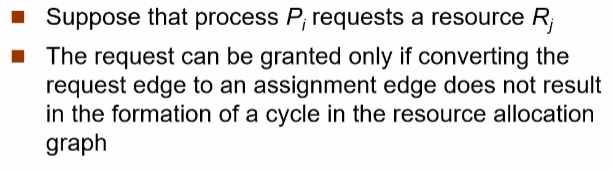
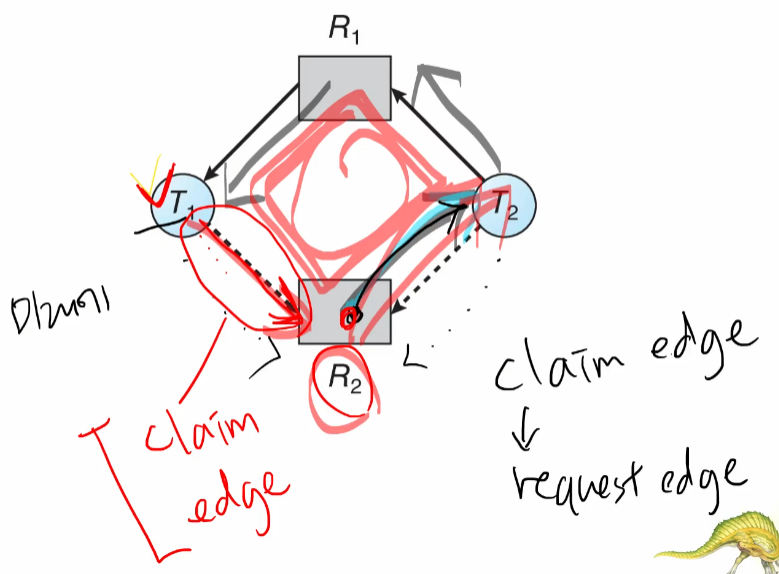
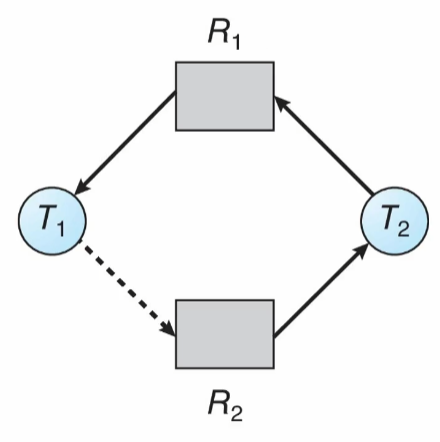
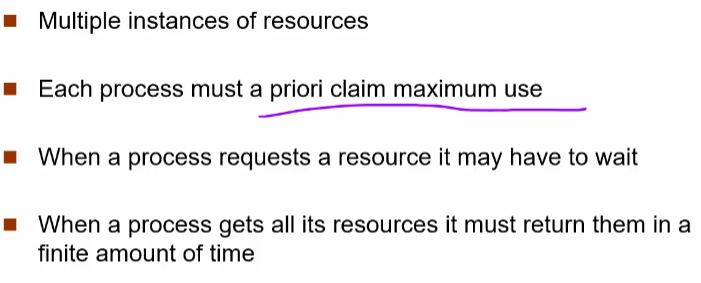
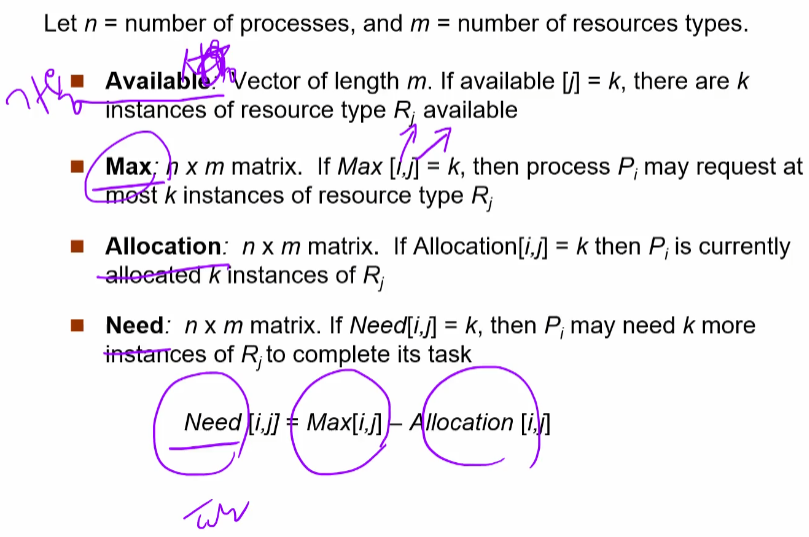
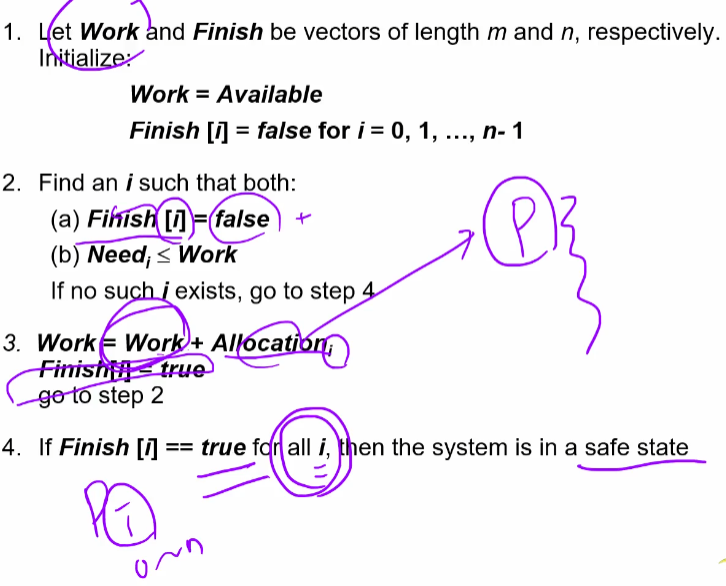
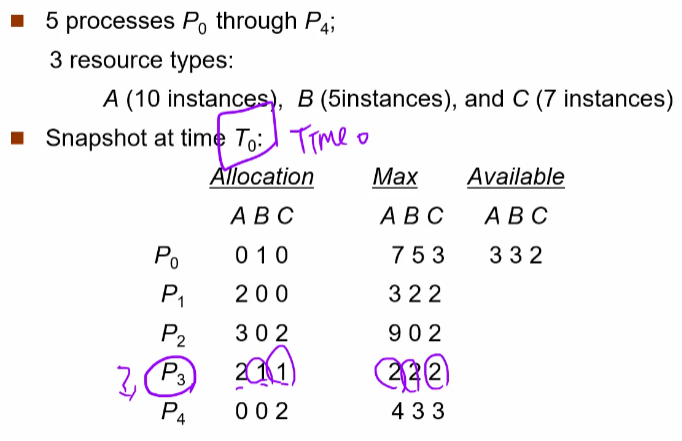
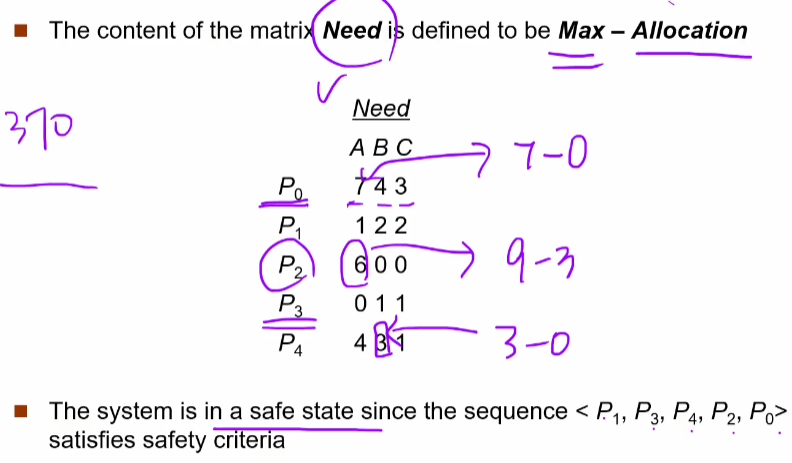
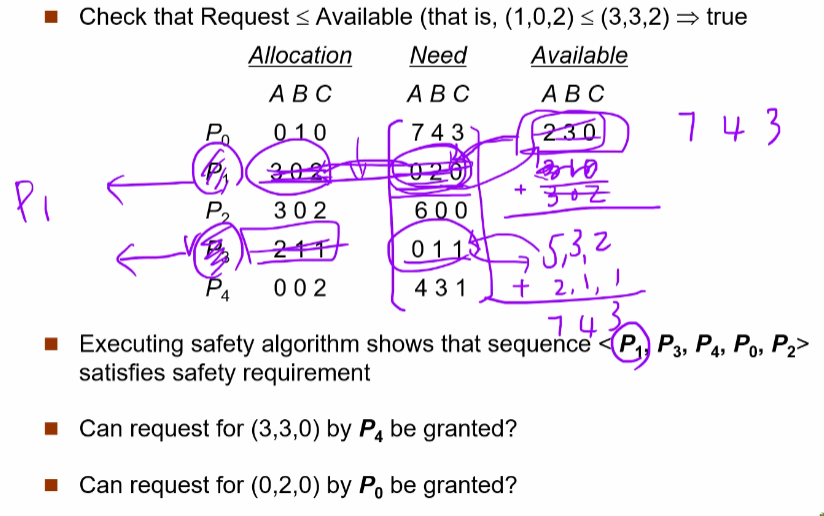
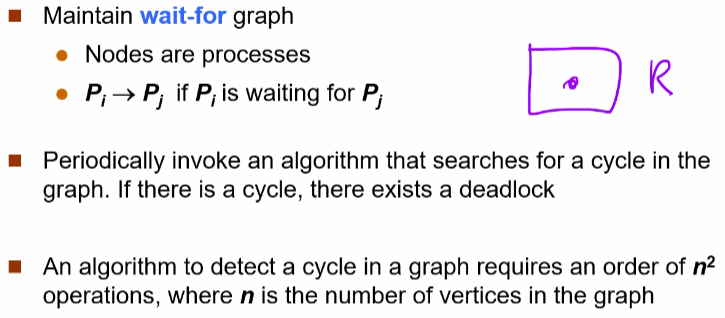
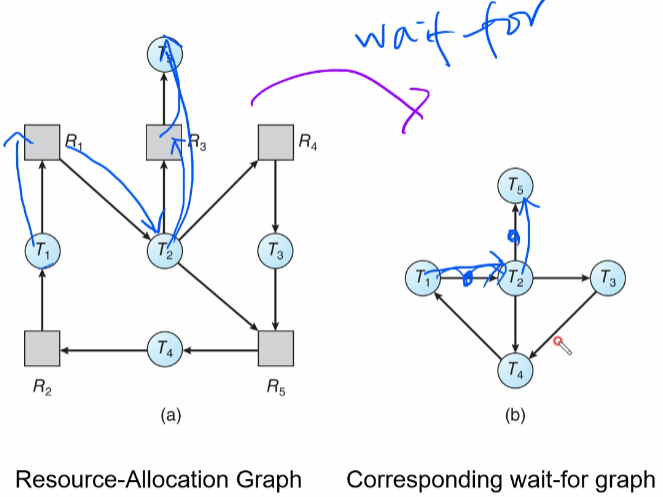
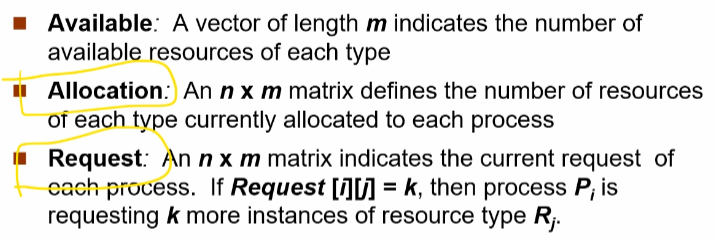
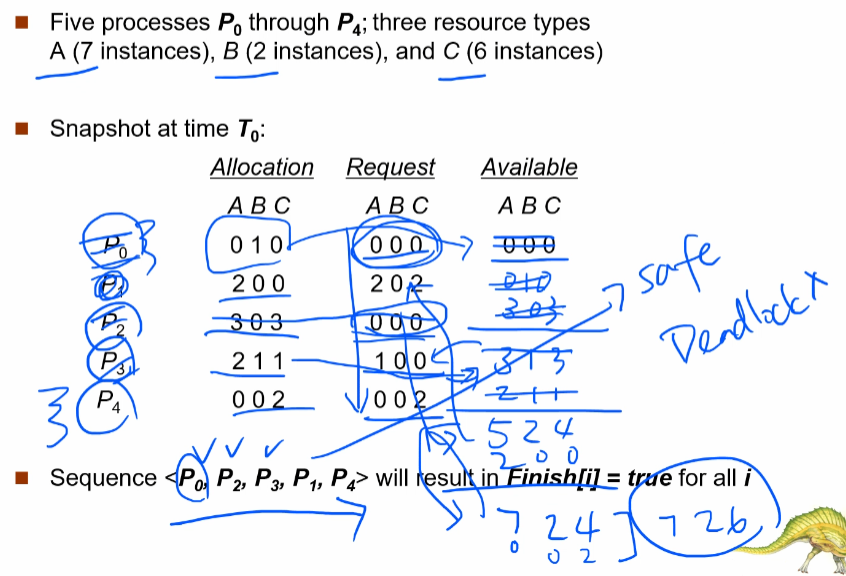
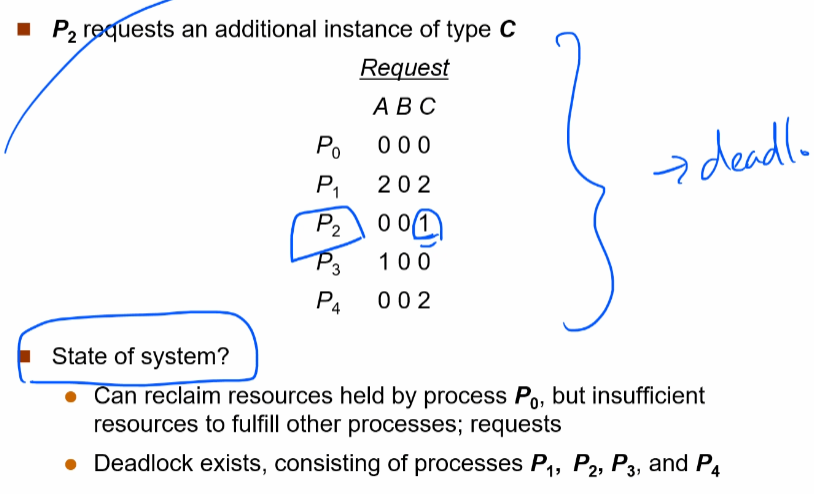
Chapter 8: Deadlocks; 교착상태

1. Deadlock Characterization; 교착 상태 특성
   1. Necessary Conditions; 필요 조건들  
      교착 상태는 한 시스템에 다음 네 가지 조건이 동시에 성립될 때 발생할 수 있다.
      1. Mutual exclusion: only the one process at a time can use a resource  
         상호 배제: 최소한 하나의 자원이 비공유 모드로 점유되어야 한다. 비공유 모드에서는 한 번에 한 프로세스만이 그 자원을 사용할 수 있다.
      2. Hold and wait: a process holding at least one resource is waiting to acquire additional resources held by other processes  
         점유하며 대기: 프로세스는 최소한 하나의 자원을 점유한 채, 현재 다른 프로세스에 의해 점유된 자원을 추가로 얻기 위해 반드시 대기해야 한다.
      3. No preemption: a resource can be released only volutarily by the process holding it, after that process has completed its task  
         비선점: 자원들이 선점할 수 없어야 한다. 즉, 자원이 강제적으로 방출될 수 없고, 점유하고 있는 프로세스가 태스크를 종료한 후 그 스레드에 의해 자발적으로만 방출될 수 있다.
      4. Circular wait: there exists a set {P0, P1, …, Pn} of waiting processes such that P0 is waiting for a resource that is held by P1, P1 is waiting for a resource that is held by P2, …, Pn-1 is waiting for a resource that is held by Pn, and Pn is waiting for a resource that is held by P0  
         순환 대기: 대기하고 있는 프로세스의 집합 {P0, P1, …, Pn}에서 P0은 P1이 점유한 자원을 대기하고, P1은 P2가 점유한 자원을 대기하고, …, Pn-1은 Pn이 점유한 자원을 대기하며, Pn은 P0가 점유한 자원을 대기한다.
   2. Resource-Allocation Graph; 자원 할당 그래프  
      A set of vertices V and a set of deges E  
      이 그래프는 정점(vertex) V의 집합과 간선(edge) E의 집합으로 구성된다.
      1. V is partitioned into two types:  
         정점 V의 집합은 두 가지 유형으로 구별된다.
         1. T = {T1, T2, …, Tn}, the set consisting of all the Thread in the system  
            시스템 내의 모든 활성 스레드의 집합인 T = {T1, T2, …, Tn}
         2. R = {R1, R2, …, Rm}, the set consisting of all resource types in the system  
            시스템 내의 모든 자원 유형인 R = {R1, R2, …, Rm}
      2. request edge – directed dege Ti -> Rj  
         요청 간선 – 스레드 Ti가 자원 유형 꺼의 인스턴스를 하나 요청
      3. assignment edge – directed edge Rj -> Ti  
         할당 간선 – 자원 유형 꺼의 한 인스턴스가 스레드 Ti에 할당
      4. 자원 할당 그래프  
         
      5. 교착 상태를 갖는 자원 할당 그래프  
         
      6. 사이클이 있으면서 교착 상태가 아닌 자원 할당 그래프  
         
      7. Basic Facts
         1. If graph contains no cycles => no deadlock
         2. If graph contains a cycle =>
            1. if only one instance per resource type, then deadlock
            2. if several instances per resources type, possibility of deadlock
2. Methods for Handling Deadlocks; 교착 상태 처리 방법
   * Ensure that the system will never enter a deadlock state:  
     시스템이 결코 교착 상태가 되지 않도록 보장
     + Deadlock prevention  
       교착 상태 예방
     + Deadlock avoidance  
       교착 상태 회피
   * Allow the system to enter a deadlock state and then recover  
     시스템이 교착 상태가 되도록 허용한 다음에 복구한다. (Deadlock detection and recover)
   * Ignore the problem and pretend that deadlocks never occur in the system.  
     문제를 무시하고, 교착 상태가 시스템에서 절대 발생하지 않는 척한다. (Ignorance)
3. Deadlock Prevention; 교착 상태 예방  
   Invalidate one of the four necessary conditions for deadlock:  
   데드락의 네 가지 조건 중 최소한 하다가 성립하지 않도록 보장
   1. Mutual Exclusion – not required for sharable resources (e.g., read-only files); must hold for non-sharable resources  
      상호 배제 – 상호 배제 조건이 성립되어야 한다. 즉 적어도 하나의 자원은 공유가 불가능한 자원이어야 한다. 반면에, 공유 가능한 자원들은 배타적인 접근을 요구하지 않으며, 따라서 교착 상태에 관련될 수 없다. 프로세스는 공유 가능한 자원을 위해 대기할 필요가 없다.
   2. Hold and Wait – must guarantee that whenever a process request a resource, it does not hold any other resources  
      점유하며 대기 - 시스템에서 점유하며 대기 조건이 발생하지 않도록 하려면 스레드가 자원을 요청할 때마다 다른 자원을 보유하지 않도록 보장해야 한다.
      1. Require process to request and be allocated all its resources before it begins execution, or allow process to request resources only when the process has none allocated to it.  
         프로세스가 자원을 전혀 갖고 있지 않을 때만 자원을 요청할 수 있도록 허용한다. 프로세스는 일부 자원을 요청하고 사용할 수도 있다. 프로세스가 추가의 자원을 요청할 수 있으려면, 자신에게 할당된 모든 자원을 반드시 먼저 방출해야 한다.
      2. Low resource utilization; starvation possible  
         낮은 자원 이용률; 기아가 발생할 수 있다.
   3. No Preemption; 비선점 –
      1. If a process that is holding some resources requests another resource that cannot be immediately allocated to it, then all resources currently being held are released
      2. Preempted resources are added to the list of resources for which the process is waiting
      3. Process will be restarted only when it can regain its old resources, as well as the new ones that it is requesting
   4. Circular Wait – impose a total ordering of all resource types, and require that each process requests resources in an increasing order of enumeration
4. Deadlock Avoidance; 교착 상태 회피  
   Basic Facts  
   
   1. Safe State; 안전 상태  
      
   2. Avoidance Algorithms  
      
   3. Resource-Allocation Graph Algorithm; 자원 할당 그래프 알고리즘  
      
      1. 교착 상태 회피를 위한 자원 할당 그래프  
         
      2. 불안전 상태의 자원 할당 그래프  
         
   4. Banker’s Algorithm; 은행원 알고리즘  
      
      1. Data Structures of for the Banker’s Algorithm  
         
      2. Safety Algorithm  
         
      3. Example of Banker’s Algorithm  
           
           
         Example: P1 Request (1, 0, 2)  
         
5. Deadlock Detection; 교착 상태 탐지
   1. Single Instance of Each Resource Type  
      
      1. (a) 자원 할당 그래프 (b) 대응되는 대기 그래프  
         
   2. Several Instance of a Resource Type  
      
      1. Example of Detection Algorithm  
           
         
6. Recovery from Deadlock: Process Termination;  
   