CLASS EXERCISE

1. Given the following resource allocation graph:
   * Resources: R1 (1 instance), R2 (2 instances), R3 (1 instance)
   * Threads:
     + T1 holds R1 and requests R2.
     + T2 holds one instance of R2 and requests R3.
     + T3 holds R3 and requests R1.  
       Construct the wait-for graph and determine if a deadlock exists. If yes, identify the deadlocked threads and propose two recovery methods (e.g., process termination or resource preemption) with pros and cons for each.
2. In a database system, transactions T1 and T2 operate as follows:
   * T1 locks Table A and then requests Table B.
   * T2 locks Table B and then requests Table A.  
     Explain how the four necessary conditions for deadlock manifest in this scenario. Propose a solution using deadlock avoidance (e.g., timestamp ordering) and another using deadlock prevention (e.g., requiring all locks to be acquired at once). Compare their practicality in high-throughput systems.

Explain how the operating system manages these transitions to ensure efficient CPU utilization and system responsiveness

Provide a real world example where a process might move back and fourth between the running and waiting state multiple times

Draw and describe the detailed swqueence of state transitions that a process undergo from creation to termination including all possible paths.