1. Draw a resource allocation graph that shows:
    a. A cycle in the graph is both necessary and sufficient condition for the existence of dead locks.
    b. A cycle in the graph is a necessary but not a sufficient condition for the existence of deadlock.
    c. A cycle exists. However, there is no deadlock
2. Identify how each of the following entities deals with a deadlock.
   Note that we can generally deal with deadlocks by ignoring the problem, preventing/avoiding it, or by detecting it then recover from it.
    a. Linux based operating systems
    b. Kernel developer
    c. Application developer
    d. Databases
3. Consider the following snapshot of a system:

	Allocation				Max			
	A	B	C	D	A	B	C	D
T_0	1	2	0	2	4	3	1	6
T_1	0	1	1	2	2	4	2	4
T_2	1	2	4	0	3	6	5	1
T_3	1	2	0	1	2	6	2	3
T_4	1	0	0	1	3	1	1	2

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

    a. $Available = (2, 2, 2, 3)$
    b. $Available = (4, 4, 1, 1)$
    c. $Available = (3, 0, 1, 4)$
    d. $Available = (1, 5, 2, 2)$

4. Consider the following snapshot of a system:

	Allocation				Max			
	A	B	C	D	A	B	C	D
T_0	3	1	4	1	6	4	7	3
T_1	2	1	0	2	4	2	3	2
T_2	2	4	1	3	2	5	3	3

T_3	4	1	1	0	6	3	3	2
T_4	2	2	2	1	5	6	7	5

Available = $(2, 2, 2, 4)$

Answer the following questions using the banker's algorithm:

a. Illustrate that the system is in a safe state by demonstrating an order in which the threads may complete.
b. If a request from thread T_4 arrives for $(2, 2, 2, 4)$, can the request be granted immediately?
c. If a request from thread T_2 arrives for $(0, 1, 1, 0)$, can the request be granted immediately?
d. If a request from thread T_3 arrives for $(2, 2, 1, 2)$, can the request be granted immediately?

5. A single-lane bridge connects the two Vermont villages of North Tunbridge and South Tunbridge. Farmers in the two villages use this bridge to deliver their produce to the neighboring town. The bridge can become deadlocked if a northbound and a southbound farmer get on the bridge at the same time. (Vermont farmers are stubborn and are unable to back up.). Note that the bridge can't holder more than one farmer at a time.

   a. Using semaphores and/or mutex locks, design an algorithm in pseudocode that prevents deadlock.
      Note: Do not be concerned about starvation (the situation in which northbound farmers prevent southbound farmers from using the bridge, or vice versa).
   b. Design a monitor that prevents both deadlock and starvation. Notes:
      - The bridge cannot hold more than one farmer. Thus, no farmer could enter the bridge while another is crossing it.
      - Northbound and southbound farmers should take turns in crossing the bridge. Two successive farmers from the same village could cross the bridge only if no farmers from the other village waiting for crossing the bridge

6. At an instant, the resource allocation state in a system is as follows:
   - 4 processes P1–P4
   - 4 resource types: R1–R4
   - R1 (5 instances), R2 (3 instances), R3 (3 instances), R4 (3 instance)

   Snapshot at time T_0:

	Allocation				Request				Available			
	R1	R2	R3	R4	R1	R2	R3	R4	R1	R2	R3	R4
P1	0	0	1	0	2	0	0	2	2	1	1	2
P2	2	0	0	1	1	3	0	1				
P3	0	1	1	0	2	1	1	0				
P4	1	1	0	0	4	0	3	1				

Run the deadlock detection algorithm and test whether the system is deadlocked or not. If it is, identify the processes that are deadlocked.

7. In Section 8.5.4, we described a situation in which we prevent deadlock by ensuring that all locks are acquired in a certain order. However, we also point out that deadlock is possible in this situation if two threads simultaneously invoke the `transaction()` function. **Fix the `transaction()` function to prevent deadlocks.**