1. Deadlock is a situation where two or more processes are waiting indefinitely for resources held by each other.
2. A deadlock will remain permanently unless some external action is taken to recover from it. Processes will not voluntarily release resources they are holding if they are perpetually waiting on other resources to become available.
3. No, deadlock cannot happen with only one process. Deadlock requires at least two processes holding resources and waiting for resources held by the other process.
4. The matrix algorithm for deadlock detection with multiple resource types:

* Mark processes that have been allocated all their maximum resources as "finish".
* Find an unmarked process P that can still request resources. Mark P as "start".
* If no such process exists, the system is deadlocked. -Else if all processes are marked "start" or "finish", the system is safe.
* Else repeat the algorithm to check next process.

1. An unsafe state is one where the system cannot ensure resources will be available for every process to complete. This may lead to deadlock.
2. Spooling keeps processes from waiting on scarce resources by having them wait on a spooler. The spooler allocates resources when available and prevents deadlock.
3. For a safe state, every sequence of resource requests should be satisfiable. Let's try x = 0, 1, 2:

* x = 0: P1 needs 1 more R1 but only 0 available - unsafe
* x = 1: Sequence P3, P4, P1, P2 is satisfiable - safe state
* x = 2: Sequence P3, P4, P2, P1 is satisfiable - safe state

So the minimum x for a safe state is 1.

1. Picture 1 is unsafe because P1 and P2 are deadlocked waiting for each other's resources.
2. The processes P1 and P4 are deadlocked. P1 is holding R2 which P4 needs, while P4 is holding R3 which P1 needs. They will wait indefinitely.
3. Recovery methods:

a) Preempt R3 from P4 and give it to P1.

b) Rollback P1 to release R2 and rollback P4 to release R3.

c) Kill either P1 or P4 to break the deadlock.