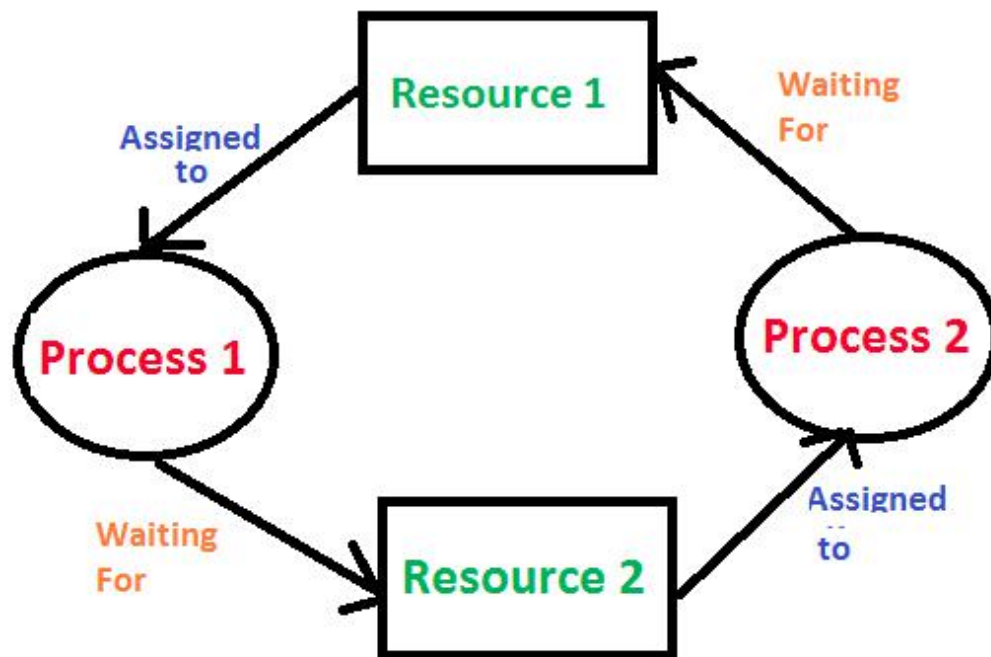


Mutual Exclusion: One or more than one resource are non-shareable (Only one process can use at a time)

Hold and Wait: A process is holding at least one resource and waiting for resources.

No Preemption: A resource cannot be taken from a process unless the process releases the resource.

Circular Wait: A set of processes are waiting for each other in circular form.



Methods for handling deadlock

There are three ways to handle deadlock

1) **Deadlock prevention or avoidance:** The idea is to not let the system into a deadlock state.

One can zoom into each category individually, Prevention is done by negating one of above mentioned necessary conditions for deadlock.

Avoidance is kind of futuristic in nature. By using strategy of “Avoidance”, we have to make an assumption. We need to ensure that all information about resources which process will need are known to us prior to execution of the process. We use Banker’s algorithm (Which is in-turn a gift from Dijkstra) in order to avoid deadlock.

2) **Deadlock detection and recovery:** Let deadlock occur, then do preemption to handle it once occurred.

3) Ignore the problem altogether: If deadlock is very rare, then let it happen and reboot the system. This is the approach that both Windows and UNIX take.