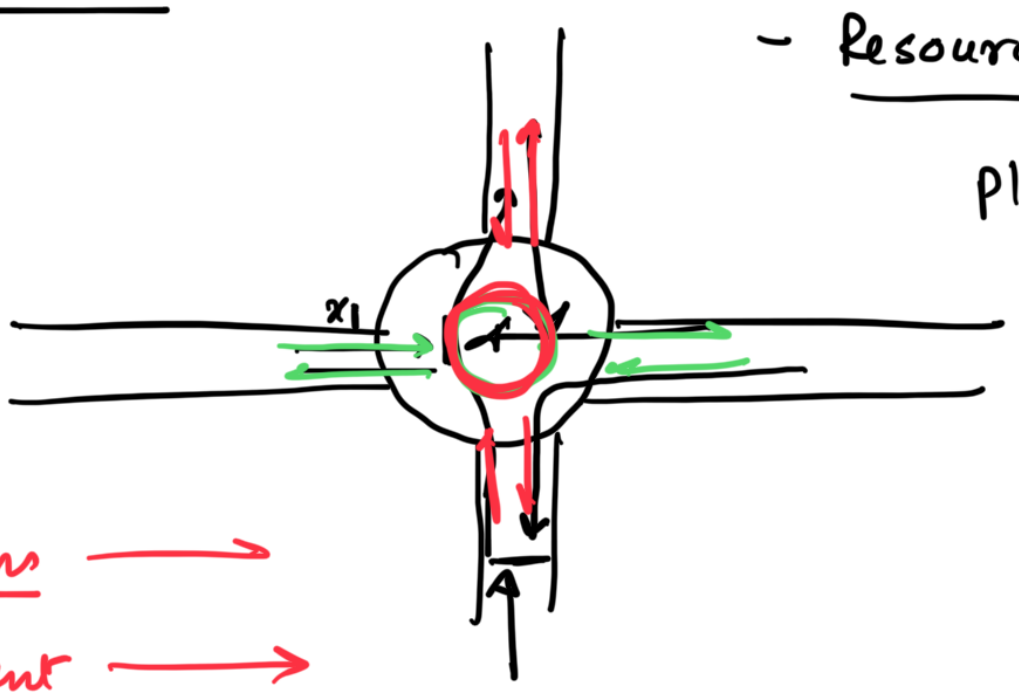
Deadlock



- Resources → imp

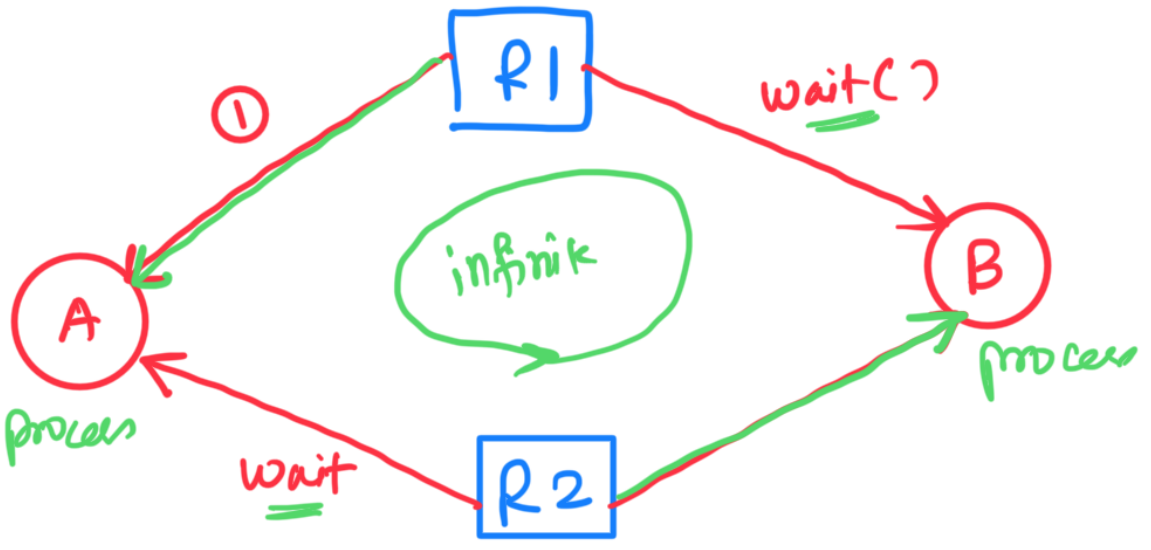
Plan our resources properly → avoid deadlock

1) Reasons →
2) Prevent →
3) Avoid →
4) Detect →
5) Recovery from Deadlock →

Deadlock - A process in OS uses different resources & uses resources in following ways

Request a resource	→	Use the resource	→	Release the resource

— Situation where a set of process are blocked because each process is holding a resource & waiting for another resources which is acquired by some other process.



Deadlock situation

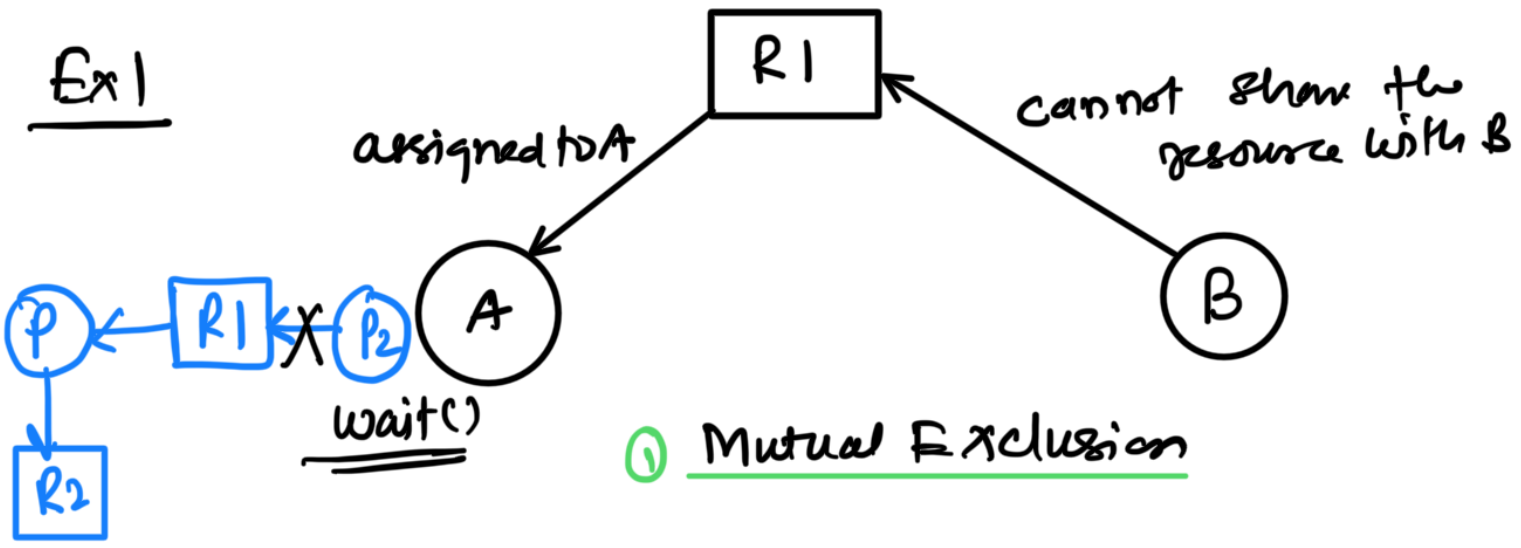
→ Request

→ Use

→ Release

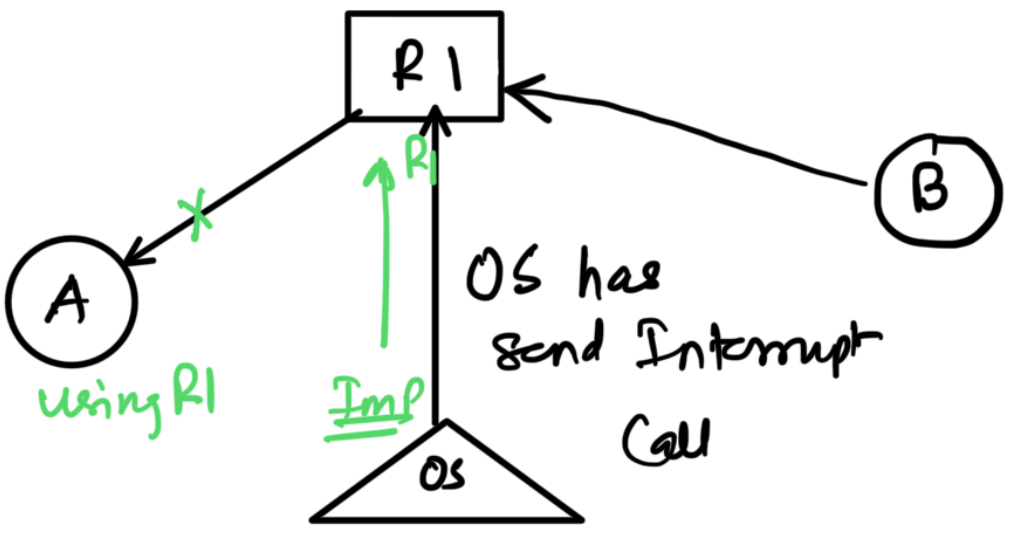
} System model for Deadlock

Ex 1



① Mutual Exclusion

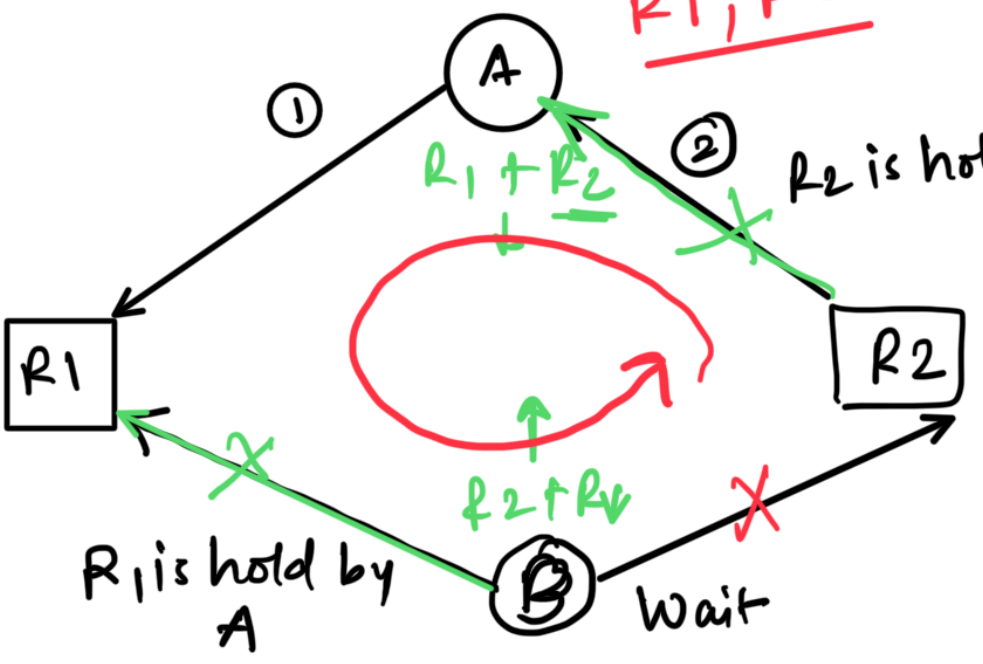
Ex 2



② Hold and wait

③ No preemption

Ex 3 :

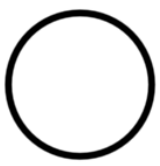


④ Circular wait

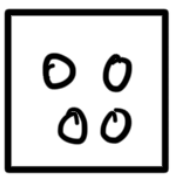
Starvation → Waiting (priority ↑ handle) (finite)

Deadlock → block the process. (infinite)

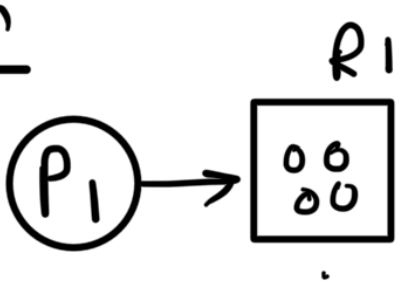
Resource - Allocation Graph



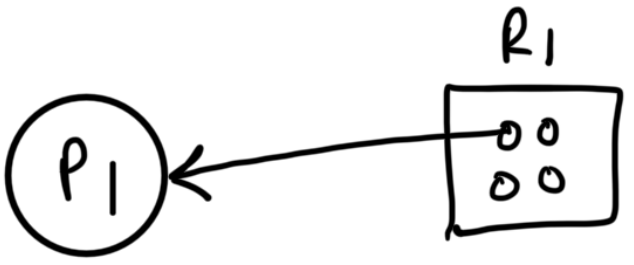
process



Resource with 4 instance

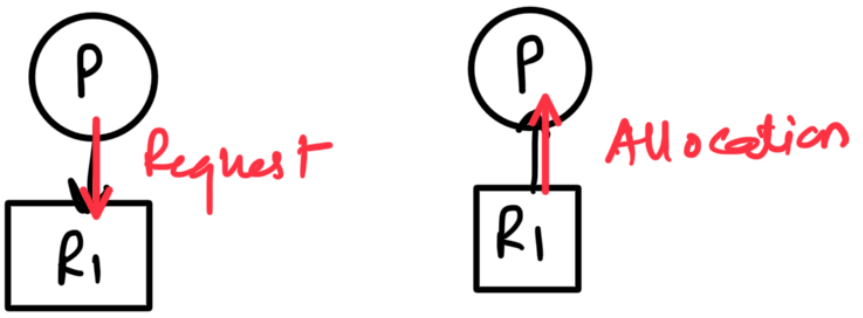


P1 is requesting instance of R1

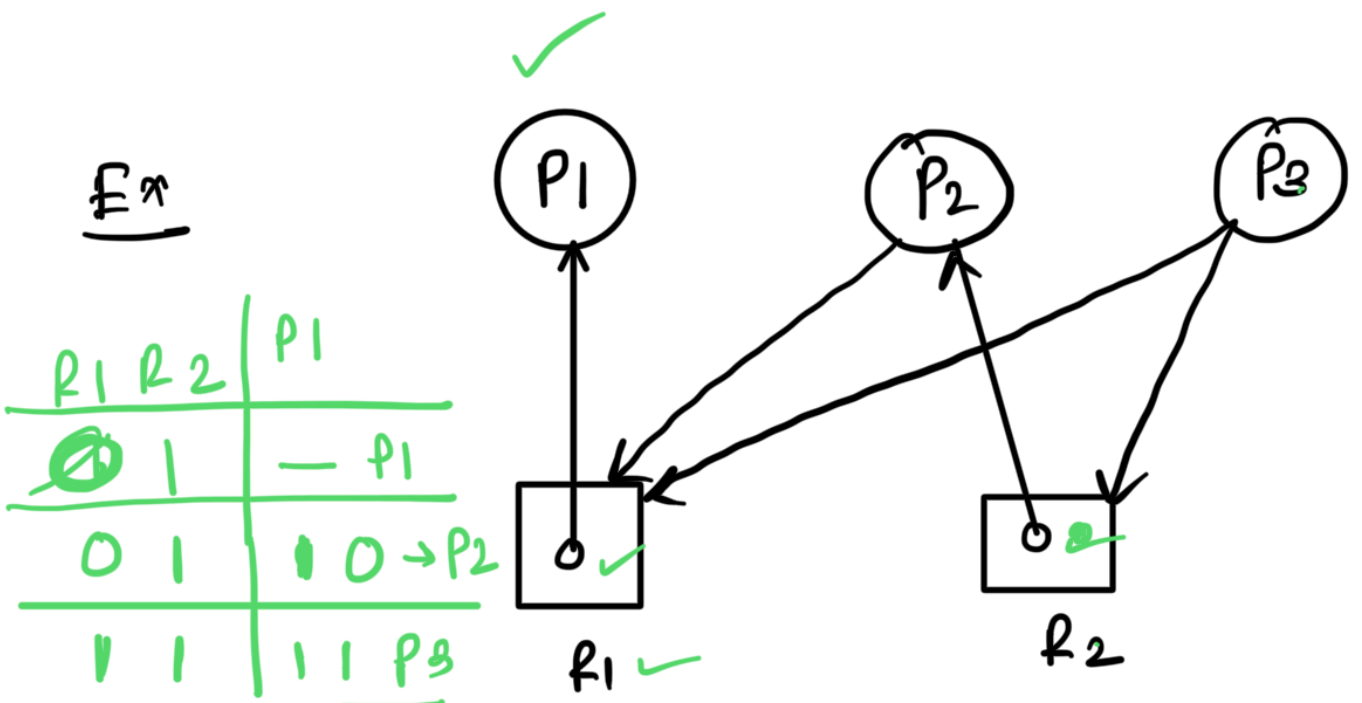


P1 is holding an instance of Resource R1

Ex



Ex



R1	R2	
0	1	— P1
0	1	0 → P2
1	1	1 P3

	Allocation		Request		
	R1	R2	R1	R2	
P1	1	0	0	0	→ No
P2	0	1	1	0	→ No

P3 0 0

1 1

Allocation (R_1 R_2)
(0 0)

P1 execution → (1 0)

P2 execution → (1 1) ✓

→ No Deadlock

P1 → P3 ✗ → handle

P1 → P2 → P3 ✓

Deadlock Handling -

1. Pre-emption
2. Roll back
3. Kill the process

Deadlock Prevention strategies -

- Mutual Exclusion
- Hold & Wait
- No Preemption
- Circular wait

} Deadlock will be avoid

Solution : Graph Allocation Method

→ Banker's Algorithm Imp

↳ help to avoid deadlock

1. No. of process to be execute.
   resource availability to be check
2. Duration of resource

3. Available resources

(1) Dining Philosophers Problem
→ Deadlock → how to handle it
↳ Solution

(2) Producer – Consumer Problem
→ Process Synchronization
→ Reader – Writer Problem

(3) Bankers Algorithm
– Deadlock handling situation