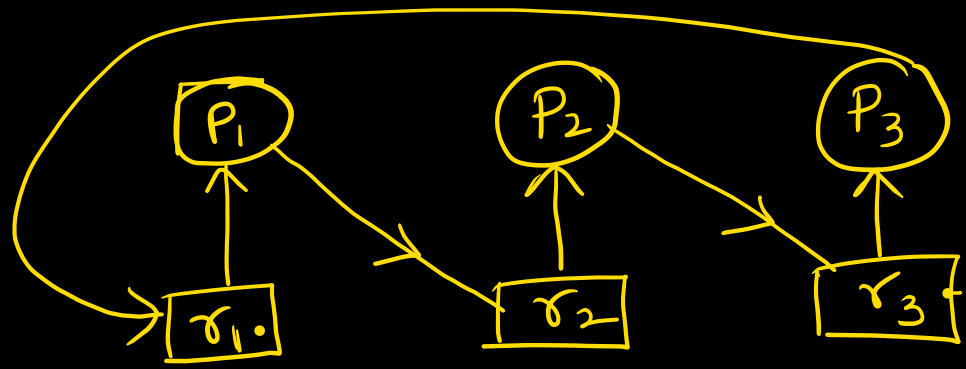
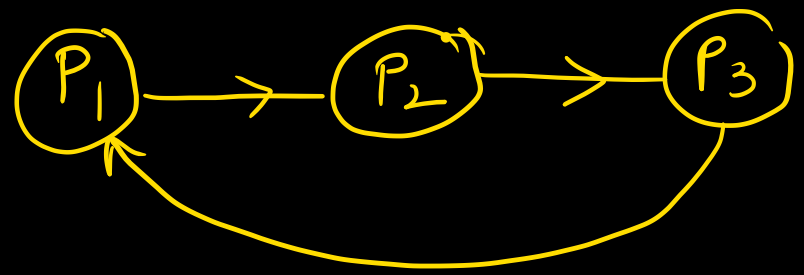
Dead locks → Dead easy

only Synchronization in diff inos.

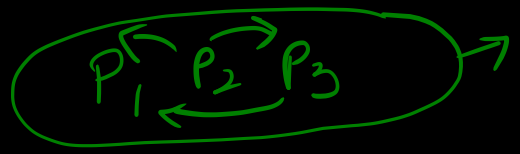
Dead lock :





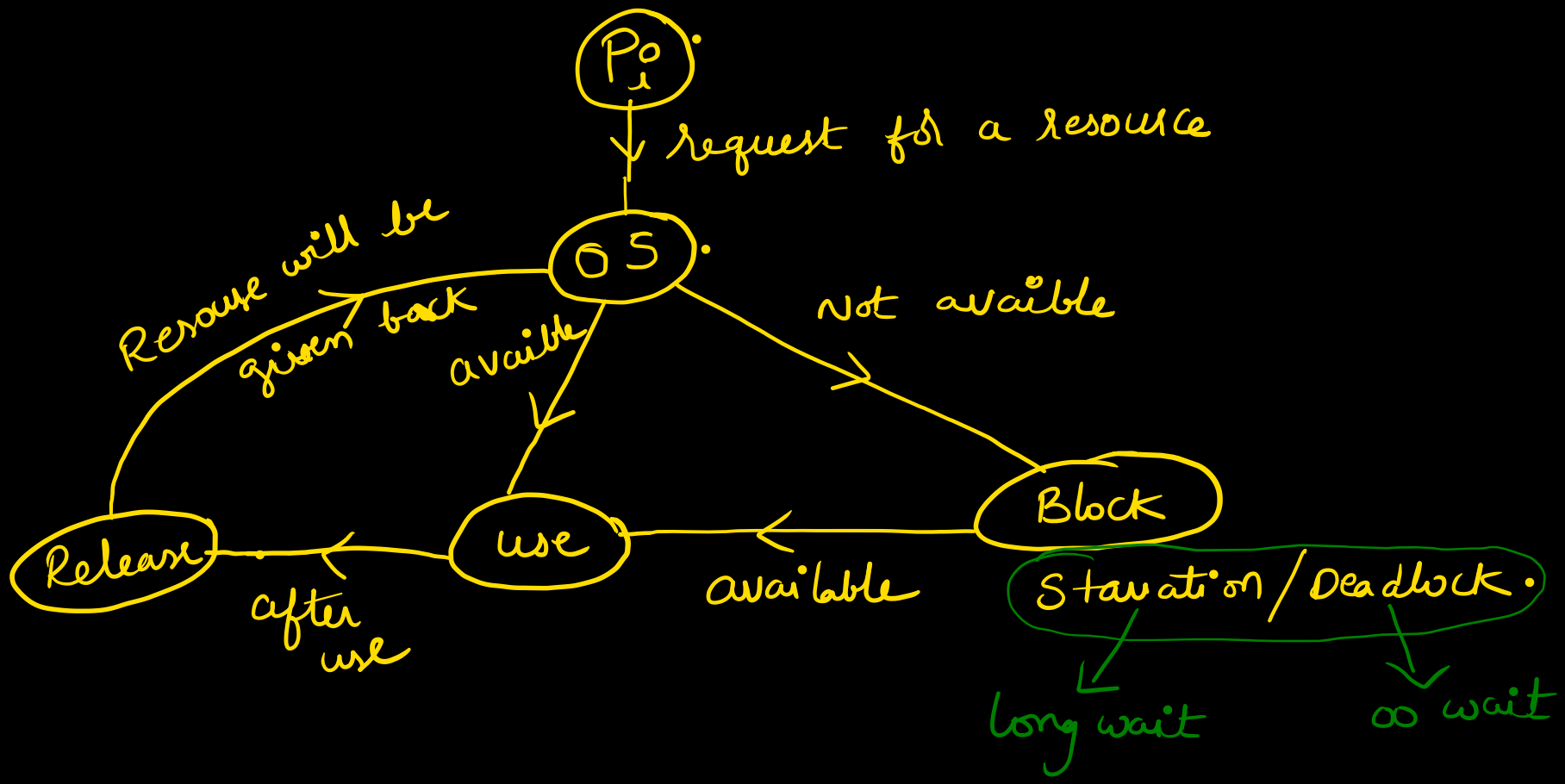
deadlock .

A set of processes are said to be in deadlock if they wait for happening of an event caused by others in the same set.



Starvation → long waiting.

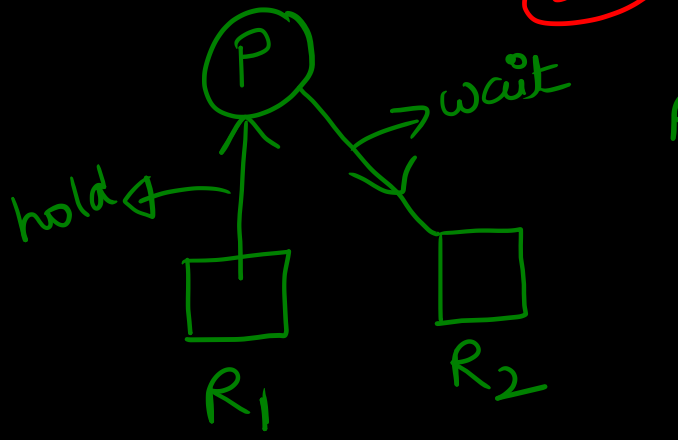
Deadlock → infinite waiting. (forever waiting)

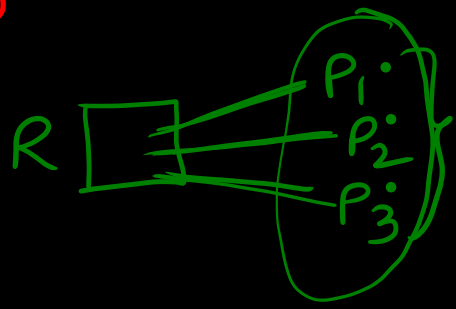

Pi
request for a resource
OS
Resource will be given back
available
not available
Release
use
Block
after use
available
Starvation / Deadlock
long wait
oo wait

Necessary conditions for DeadLock:

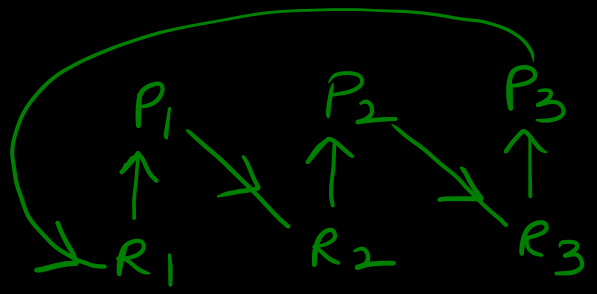
- mutual exclusion → A Resource Cannot be accessed by more than one process at the same time
- Hold and wait → A processes should wait for a rec while holding another res
- no preemption → we should not pull a resource by force.
- Circular wait → Processes should wait for each other in a circle.

DL.





No deadlock.



Circular wait.

Necessary Condition : Clouds are necessary for Rain.

Sufficient Condition : Clouds are not sufficient for Rain

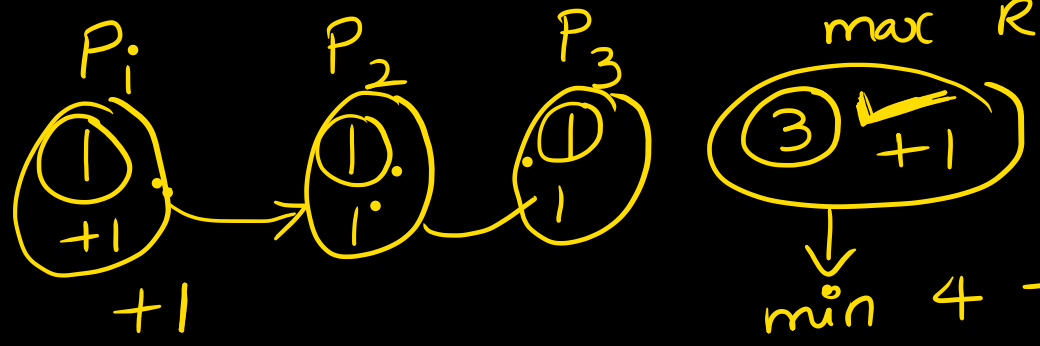
4 Conditions → necessary for DL.

which mean if there has to be DL, then 4 Cond should be true

But not sufficient

→ If 4 conditions are true, there may / may not be DL.

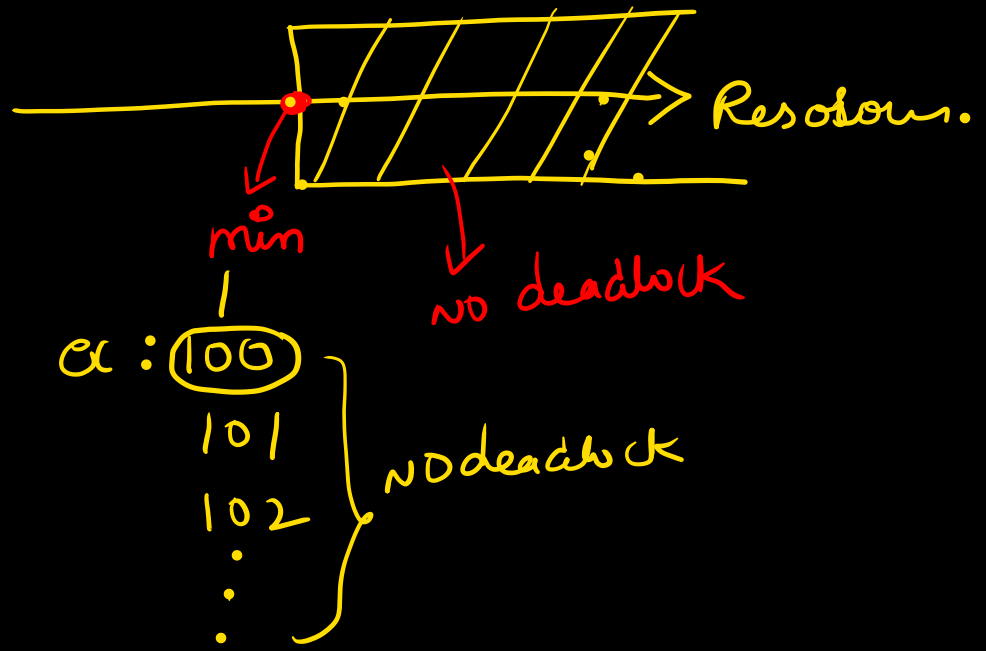
A system is having 3 user processes each requiring 2 units of resource 'R'. The minimum number of units of 'R' such that no deadlock occurs is.

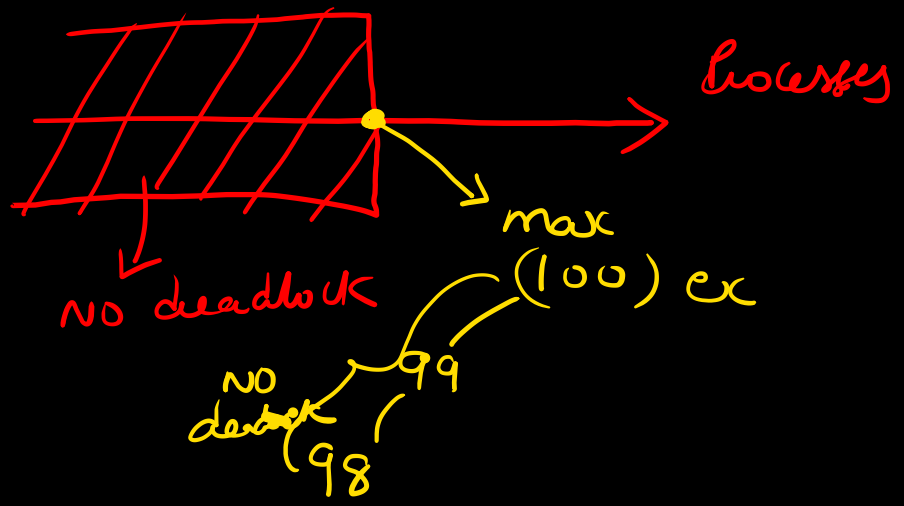


P_1	P_2	P_n
R_1	R_2	R_n

(2) ✓ min no of Resources such that no DL occur

max no of processes such that no DL occurs.





	P_1	P_2	P_4
R	2	3	4

min no of 'R' for no deadlock.

max no of R → still deadlock

P_1 P_2 P_3 ✓

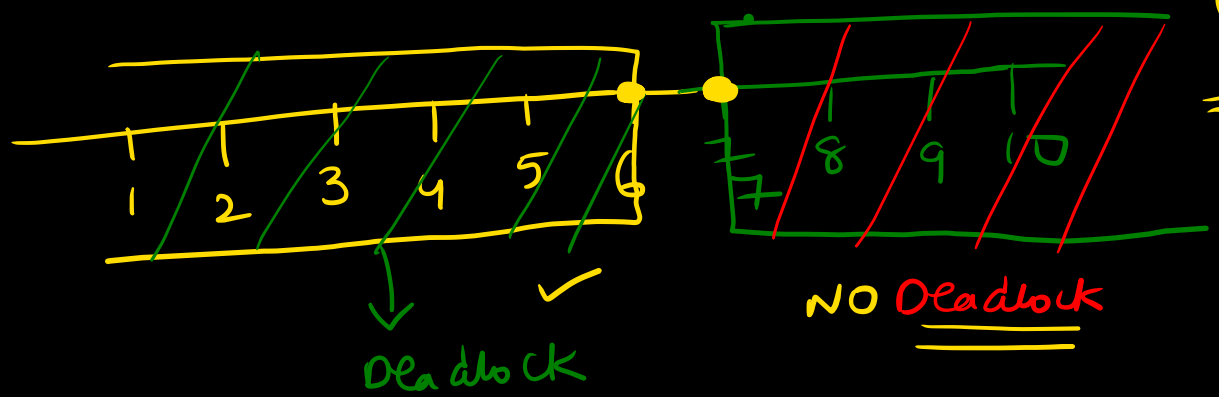
2-1 3-1 4-1

1. 2. 3.

6+1 → min 'R'

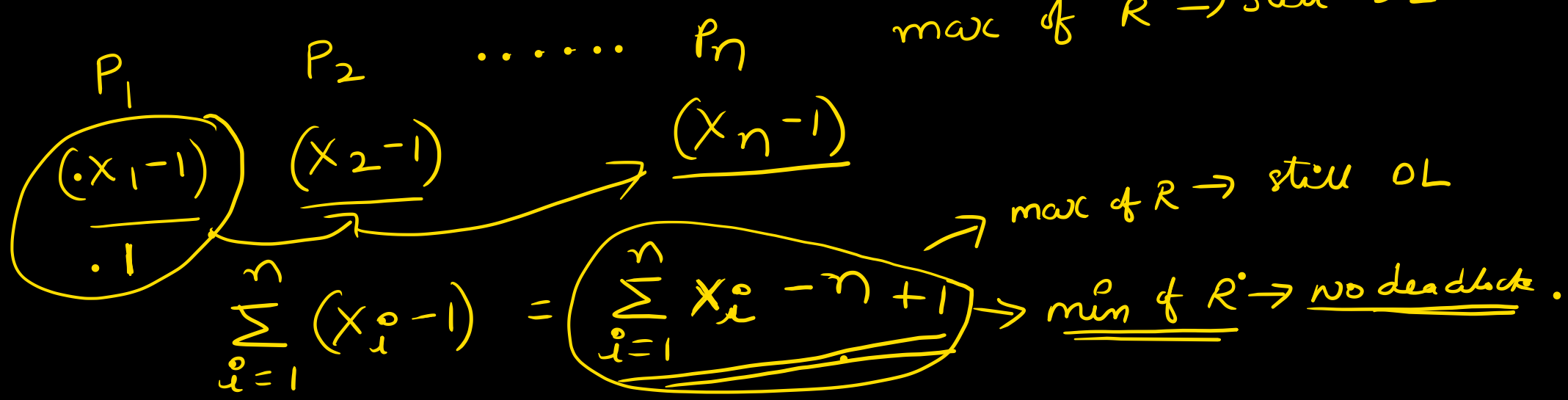
6^{+1} → max no of Re still deadlock.

7, 8, 9, 10 .... ∞.



	P_1	P_2	.....	P_n
R	x_1	x_2		x_n

min 'R' no deadlock?



There are 6 instances of a resource and every process P_i requires 2 instances of that resource to finish its execution Then how many processes can be present at max so that there will be no dead lock.

$R = 6$, $P_i \rightarrow 2$, min of processes → still there deadlock

Processes → max

Resour → min

P_1	P_2	P_3	P_4	P_5	P_6
1	1	1	1	1	1

6 → DL

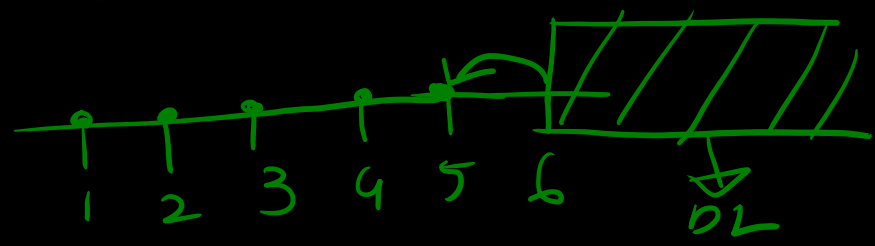
5 → NoDL.

5

6 process → deadlock.

7 process → d → DL

8



6 instances of 'R', P_i needs 3 instances max # of P no DL = ?

$P_i \rightarrow (3-1) = 2$

min.

P_1 P_2 P_3 → 3 process → DL

1 1 1

2 2 2 → 6

3 P → DL

−1

2 P → no dL.

max proc → no DL

100 instances of R, P_i → 2 instances → max # P & no DL ?

$$P_i \rightarrow (2-1) = 1.$$

$$n * 1 = 100$$

$$n = 100$$

→ min # of processes for DL

$n-1 = 99$ → max # of processes in NO DL.

100 instances, P_i → 3 instances, max # P ∄ no DL = ?

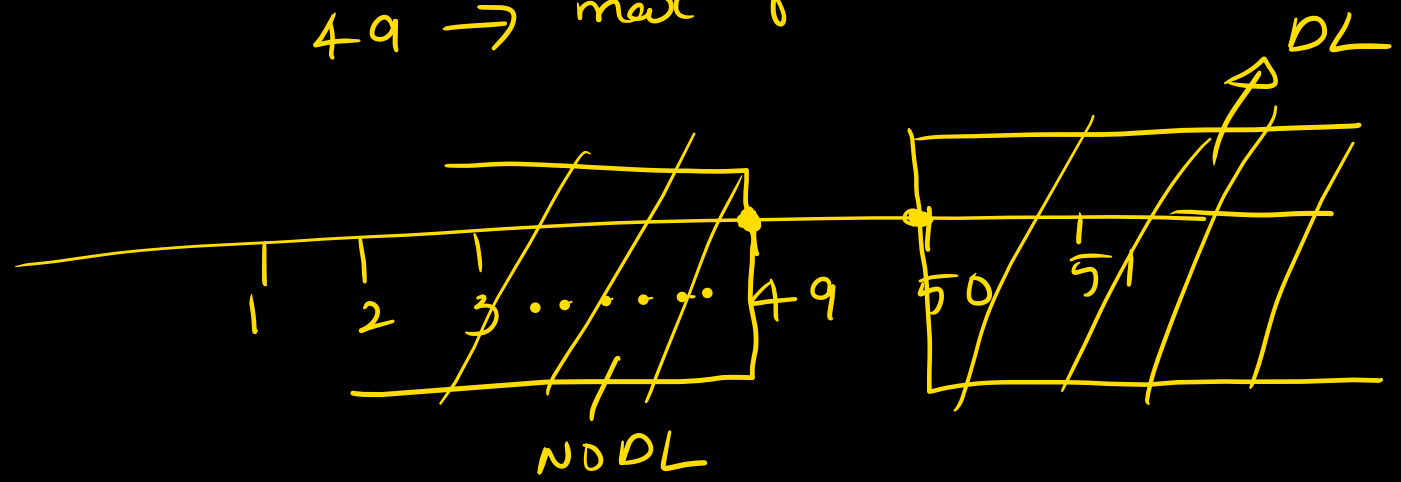
$P_i \rightarrow (3-1) = 2$

$n * 2 = 100$

$n = 50$ → min # P ∄ DL

-1

49 → max # P ∄ no DL



100 instances, P_i → 4 instances, max # of P so that no deadlock?

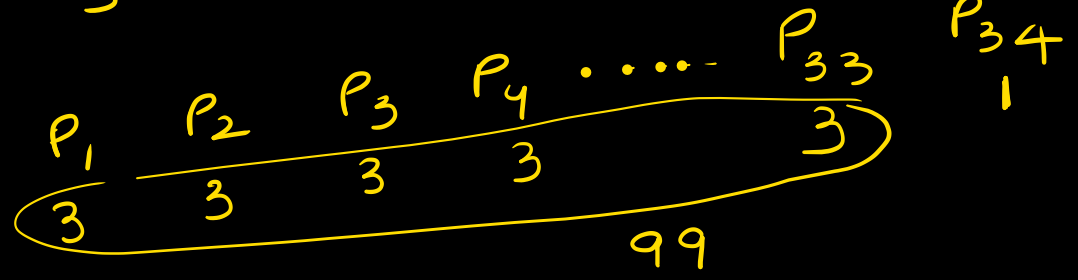
$P_i \rightarrow (4-1) = 3$

$n * 3 = 100$

$n = \frac{100}{3} = 33.3 \rightarrow 34$ ✓

n → min.

min = 33.3 → 34



34

|-1

33 → max no of process with no deadlock

10 processes, each process requires (3) instances, min # of resources such that no deadlock?.

P_1	P_2	P_3	.....	P_{10}
2	2	2		2

→ (20) → max of resources still DL

↓ +1

(21) → min # of Resources no DL.

Gate 92: A computer system has 6 tape drives, with n processes competing for them. Each process need 3 tape drives. The maximum value of 'n' for which system is DL free

$P_i \rightarrow (3) - 1 \rightarrow 2$

$n * 2 = 6$

$n = (3) \rightarrow$ min # of procs $\rightarrow$ DL

$\downarrow -1$

$(2) \rightarrow$ max # of processes $\rightarrow$ no DL

Gate 93: Consider a system having 'm' resources of the same type. These resources are shared by 3 processes A, B and C, which have peak demands 3, 4 and 6. For what value of 'm' deadlock will not occur.

m is not used

a) 7 b) 9 c) 10 ~~d) 13~~

A	B	C
1	1	1
(3 -1)	(4 -1)	(6 -1)
2	3	5

→ 10 max DL

↓ +1

(11) → min DL true

11, 12, 13, 14, 15

Gate05

Suppose n processes $P_1, P_2 \ldots P_n$ share 'm' identical resource units which can be reserved and released one at a time. The maximum resource requirement of process P_i is S_i where $S_i > 0$. Which one of the following is a sufficient condition for ensuring DL free.

a) $\forall_i \; S_i < m$ b) $\forall_i \; S_i < n$ c) $\sum_{i=1}^{n} S_i < (m+n)$ d) $\sum_{i=1}^{n} S_i < (m \times n)$

P_1	P_2	P_3	....	P_n
(S_1-1)	(S_2-1)	(S_3-1)		(S_n-1)

$$\sum_{i=1}^{n}(S_i-1) = \sum_{i=1}^{n} S_i - \sum_{i=1}^{n} 1 = \sum_{i=1}^{n} S_i - n < m$$

$$\sum_{i=1}^{n} S_i < m+n$$

Strategies for handling Deadlocks:

1) Deadlock ignorance
2) Deadlock Prevention → Before it happens → OL never happens.
3) Deadlock avoidance
4) Deadlock detection and recovery.

DL — ① in 4 years

Deadlock ignorance :

Act as if there is no deadlock.

If all processes get into deadlock ⇒ Restart the system.

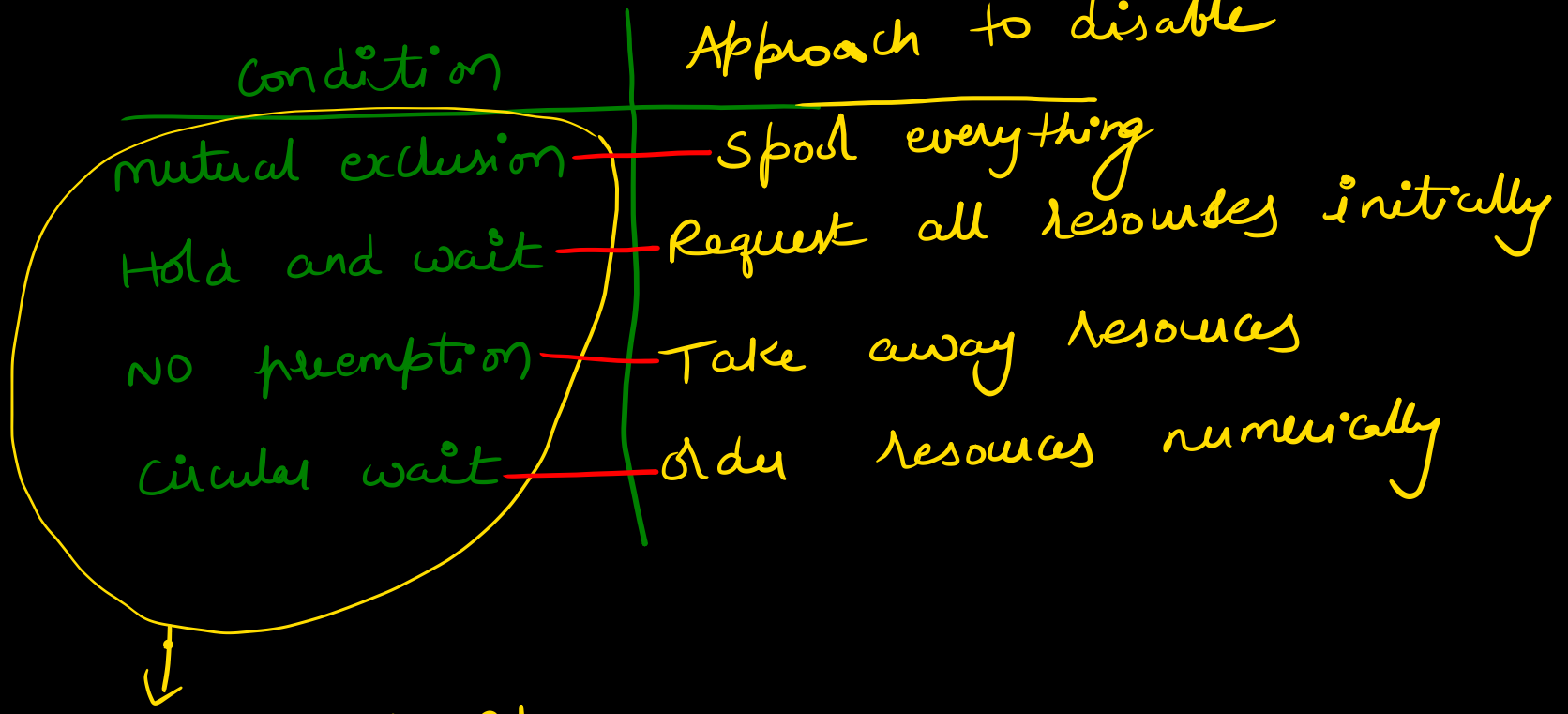
Window and Linux follow DL ignorance.

NO action is taken to prevent DL, NO detection of DL,

NO avoidance, NO recovery. Simply ignore.

Ostrich approach.

Deadlock Prevention:



necessary for DL

If we disable at least one condition, there will be no deadlock