CS & IT ENGINEERING

Operating Systems

Deadlock

Lecture No. 3

By- Dr. Khaleel Khan Sir

TOPICS TO BE COVERED

- Banker's Algorithm
- Deadlock Detection & Recovery
- Problem Solving

Deadlock Avoidance (Banker's Algo) [M.I. Resource]

$\langle A, B, C \rangle = \langle 10, 5, 7 \rangle$ Total

RQ T_0

	Allocation			Max			Available			Need		
	A	B	C	A	B	C	A	B	C	A	B	C
P0	0	1	0	7	5	3	~~3~~	~~3~~	~~2~~	7	4	3
P1	~~2~~ 3	~~0~~ 0	~~0~~ 2	3	2	2	2	3	0	~~1~~ 0	~~2~~ 2	~~2~~ 0
P2	3	0	2	9	0	2				6	0	0
P3	2	1	1	2	2	2				0	1	1
P4	0	0	2	4	3	3				4	3	1

T_0: $\langle P_1; P_3; P_4; P_0; P_2 \rangle$ ∴ Safe

T_1: P_1 :→ ✓ $Req_1 \langle 1, 0, 2 \rangle$

T_2: P_4 :→ $Req_4 \langle 3, 3, 0 \rangle$ ✗

T_3: P_0 :→ $\langle 0, 2, 0 \rangle$ ✗

Bank (Branch)

SBI

C

S.B : 20L (Max)

withdrawn : 2L (Alloc)

Balance : 18L (Need)

t_1 : C → 16L

9:30 am

Cheque

<Req.>

Banker

~~18L~~ 100L

Avail Cash

"After granting the Req/i the Resulting State Should be Safe"

(ii) Resource - Request Algorithm

Algorithm Res_Request (P_i, Req_i, $Alloc_i$, $Need_i$, Avail)

{

1. $Req_i \le Need_i$
2. $Req_i \le Avail$
3. [Assume to have Satisfied the Req_i]
   a) $Avail = Avail - Req_i$
   b) $Need_i = Need_i - Req_i$
   c) $Alloc_i = Alloc_i + Req_i$
4. Run Safety Algo
5. If System is SAFE then Grant the Req_i
   else deny the Req_i & Block the Process P_i

}

Process	Allocation	Max	Available
	A B C D	A B C D	A B C D
P0	0 0 1 2	0 0 1 2	1 5 2 0
P1	1 0 0 0	1 7 5 0	
P2	1 3 5 4	2 3 5 6	
P3	0 6 3 2	0 6 5 2	
P4	0 0 1 4	0 6 5 6	

Practice Q's:

Safe / NOT

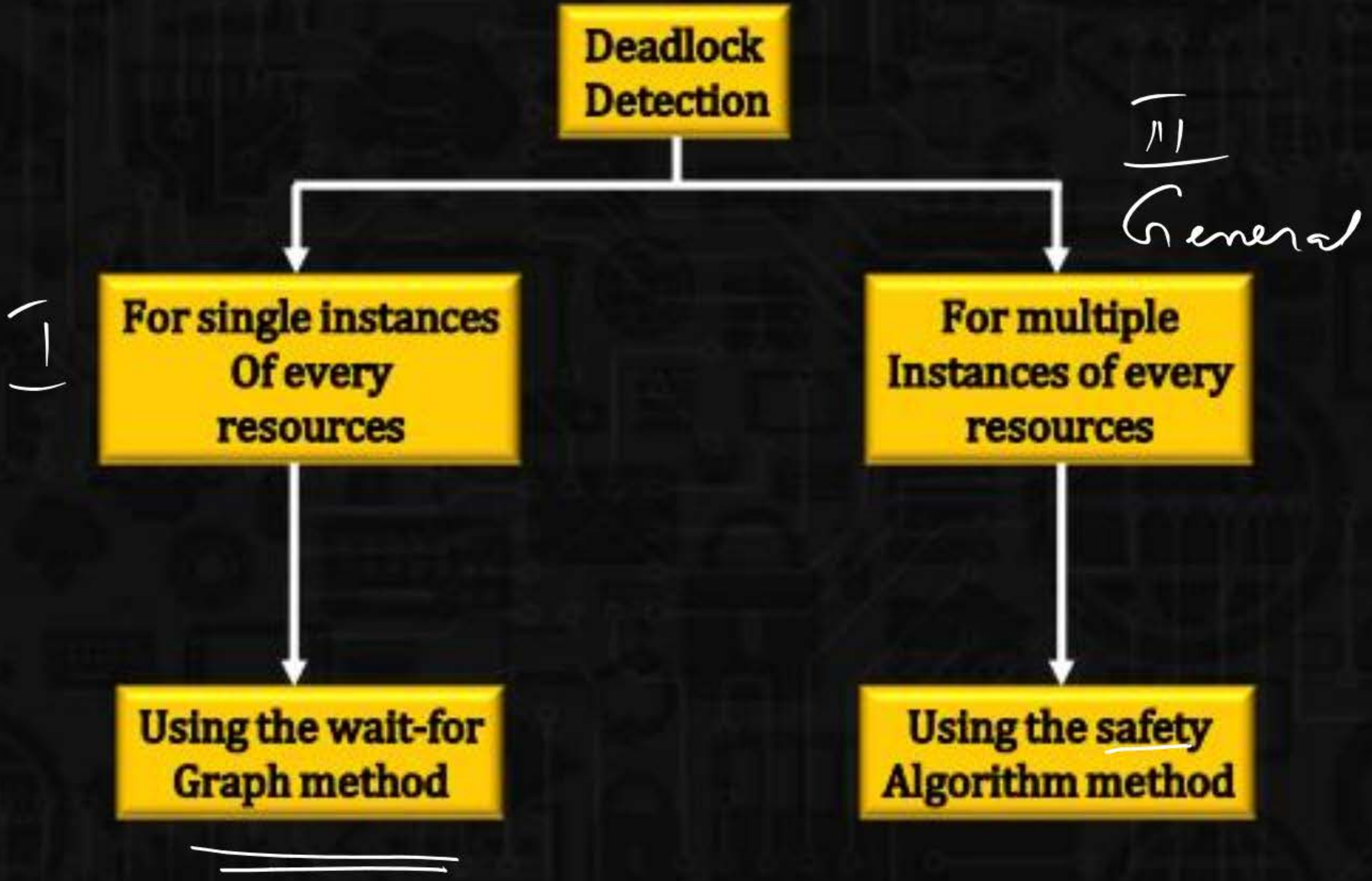
IV. Deadlock Detection & Recovery

When to activate

(apply detection Algo)

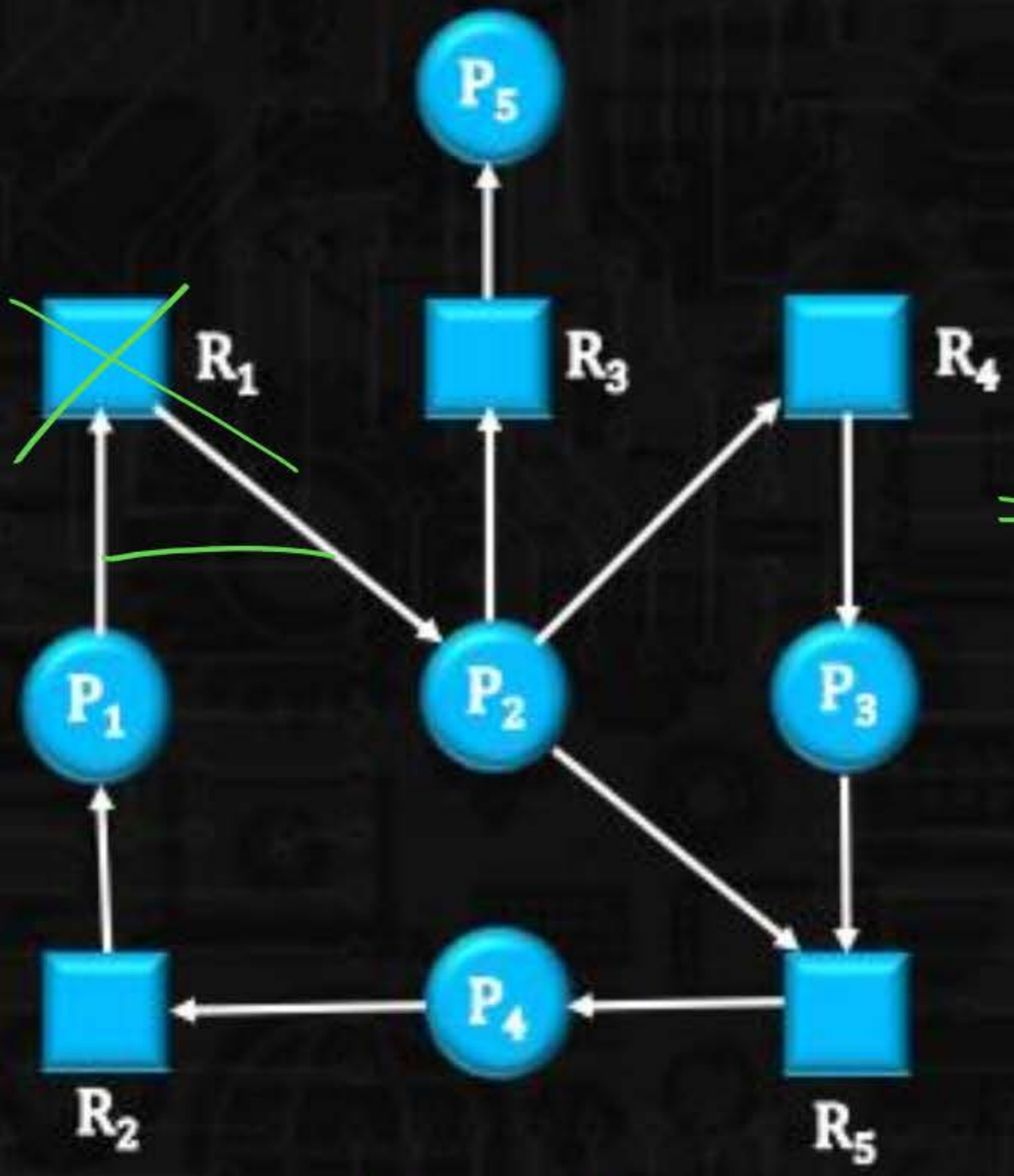
→ under utiliz. of CPU

→ Majority of Processes are Blocked


Deadlock Detection
II
General
I
For single instances Of every resources
For multiple Instances of every resources
Using the wait-for Graph method
Using the safety Algorithm method

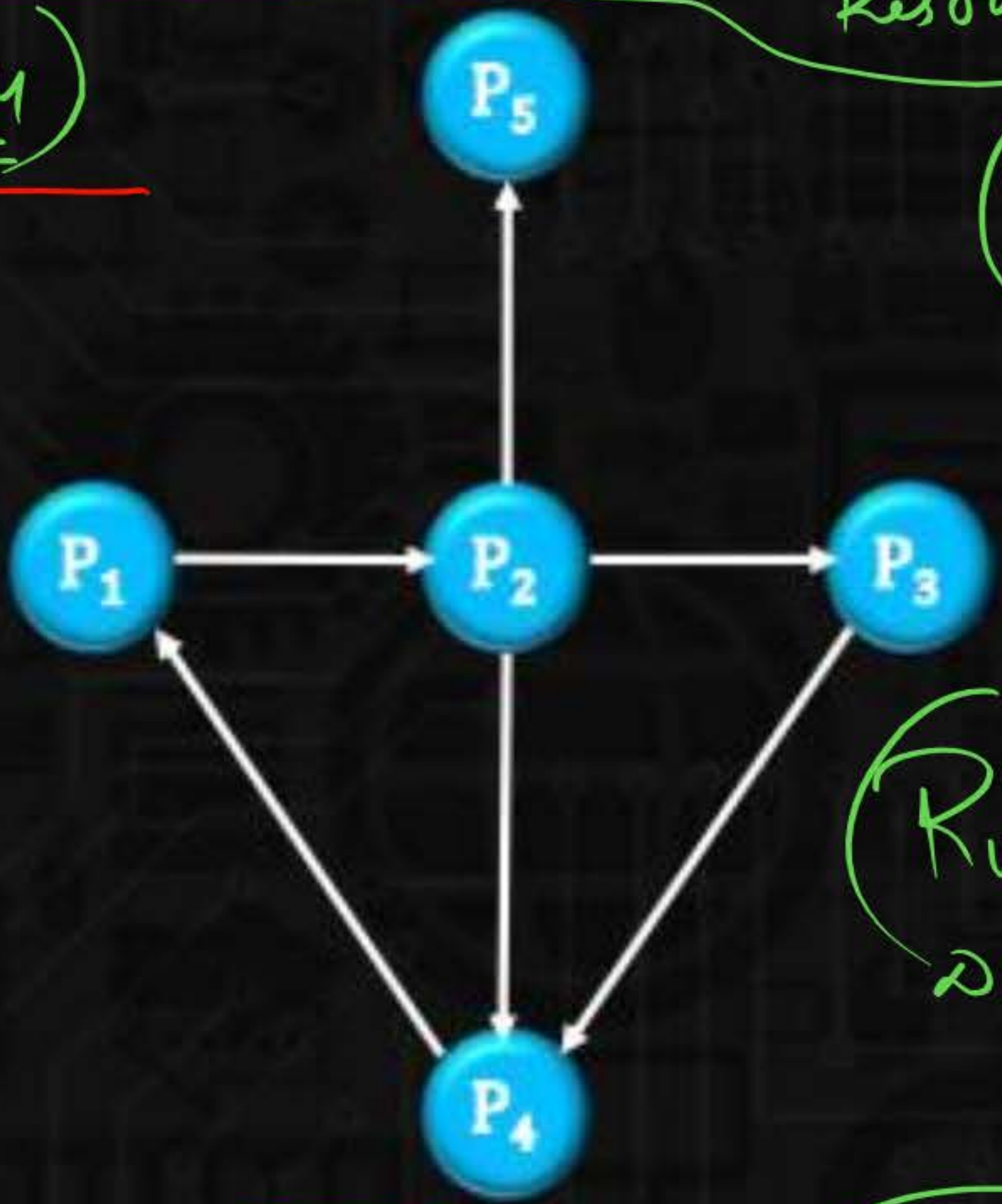
I. Deadlock Detection for Single Instance Resource

O/P : (P_1 - - P_4)



Ex: G

Wait-for graph



Safe / Deadlock

(Run cycle detection Algo)

$P_1 - P_2 - P_3 - P_4 - P_1$: Cycle → Deadlock

2) Deadlock detection with Multi-Instance Resource <Safety Algo> Approach (Safe + Deadlock)

$\langle A, B, C \rangle = \langle 7, 2, 6 \rangle$ Total

'n'	Allocation			Request			Available		
	A	B	C	A	B	C	A	B	C
P0	0	1	0	0	0	0	0	0	0
P1	2	0	0	2	0	2 ×	⟨0 1 0⟩		
P2	3	0	3	0	0	~~0~~ 1 ×			
P3	2	1	1	1	0	0 ×			
P4	0	0	2	0	0	2 ×			

$\langle P_1 \ldots P_4 \rangle$ Deadlock O/P

T_0: $\langle P_0; P_2, P_3; P_4; P_1 \rangle$ ✓ : Safe

T_1: P_2 : $\rightarrow \langle 0,0,1 \rangle$ Req

Blocked

$\langle P_0;$ (Deadlock)

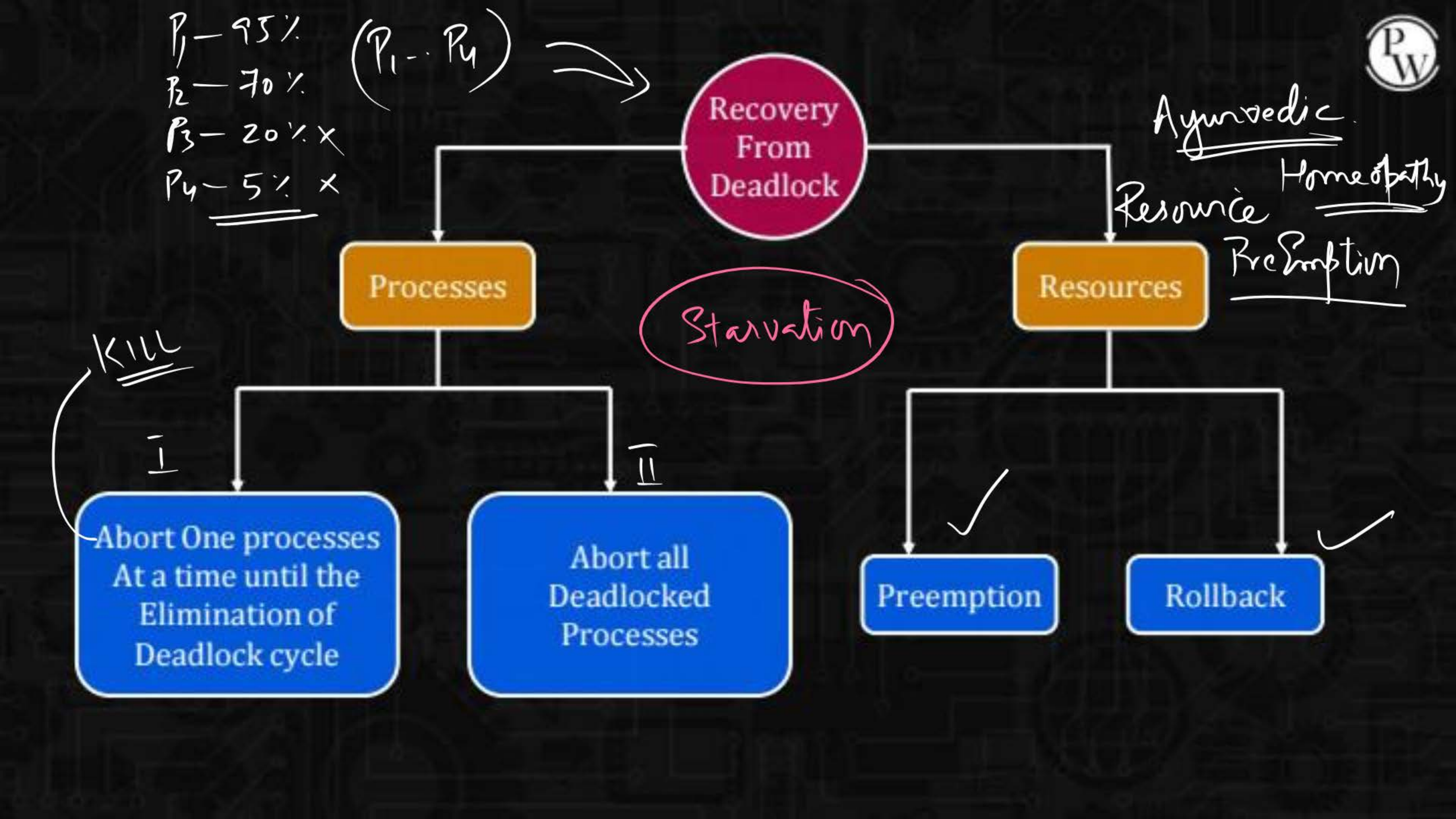
Note: Sys is safe iff the Request of all Processes are satisfiable with the avail copies in some order otherwise deadlock

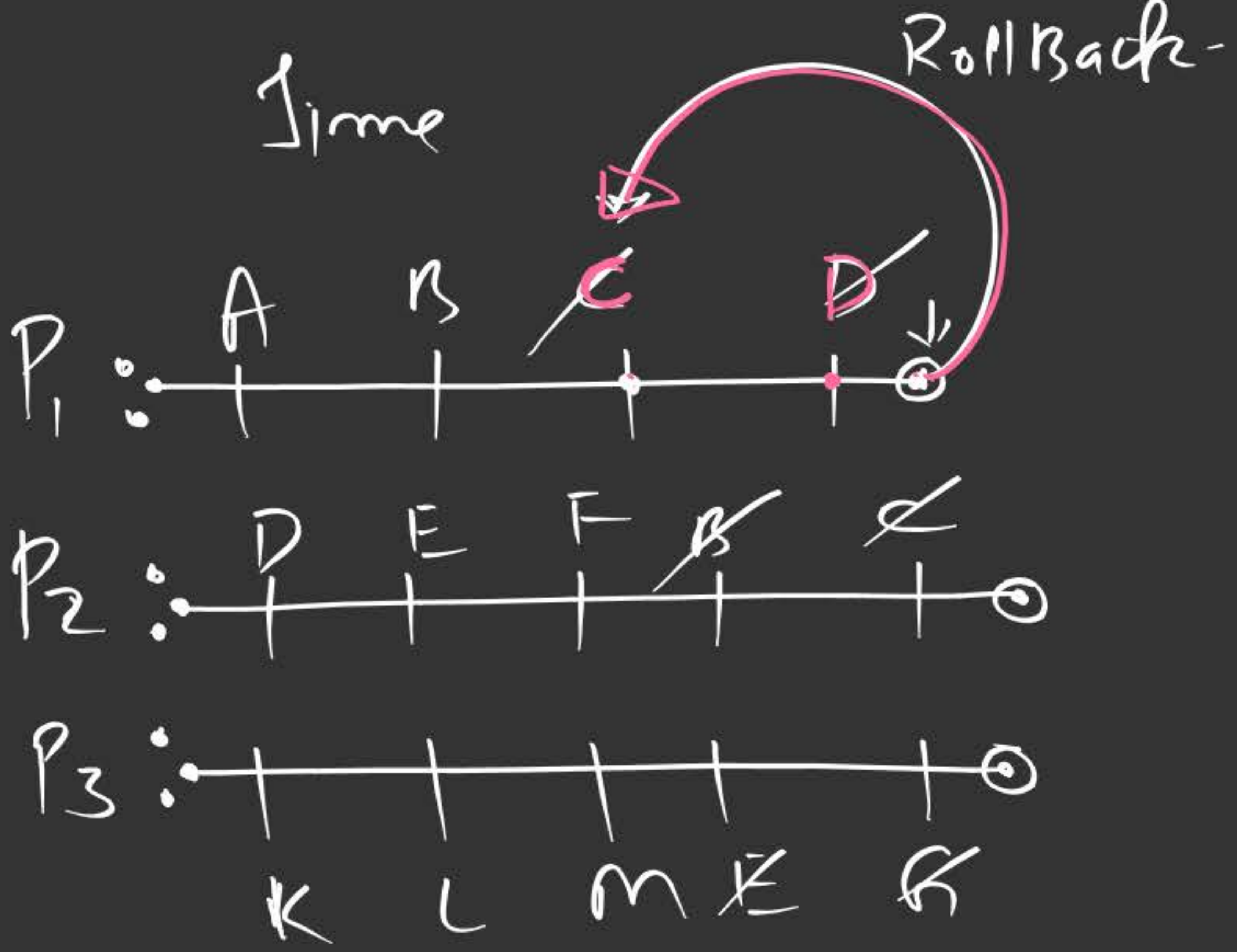
Q. Apply deadlock detection algorithm to solve the following program. There are five processes and 4 resources types.

Practice Q

	Allocation				Request				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P1	1	0	0	0	0	1	0	0	2	0	0	0
P2	0	1	0	0	0	0	1	0				
P3	0	0	1	0	0	0	0	1				
P4	0	1	0	1	1	0	0	0				
P5	0	0	0	1	0	0	0	0				

Do a step by step execution of the Dead lock detection algorithm to find the processes are in deadlock? If the system has no deadlock show the execution sequence processes?


Recovery From Deadlock
Processes
Resources
Abort One processes At a time until the Elimination of Deadlock cycle
Abort all Deadlocked Processes
Preemption
Rollback
P1 — 95%
P2 — 70%
P3 — 20% ×
P4 — 5% ×
(P1 .. P4)
Starvation
KILL
I
II
Ayurvedic
Homeopathy
Resource PreEmption



Resource PreEmption

Avail: $\langle D, C, G \rangle$

$\langle C, B, E \rangle$

1. Prevention
2. Avoidance

$\Big]\ C_1$ Proactive

3. Detection & Recovery $\Big]\ C_2$ Reactive

4. Ignorance $\Big]\ C_3$ Inactive

Conceptual Problems

① Consider a System having 'n' processes & a Single 'R' having '6' copies; Each Process Need 2 copies of R to Complete.

a) What is the Min(n) to Cause deadlock? 6

b) " " " Max(n) for deadlock freedom? 5

$R = 6$; $P_i \rightarrow 2(R)$

$P_1 - 1$
$P_2 - 1$
$P_3 - 1$
$P_4 - 1$
$P_5 - 1$
$P_6 - 1$

$P_1 - 1$
$P_2 - 11$
$P_3 - 1$
$P_4 - 1$
$P_5 - 1$

Blocked

Q2) $n = 3$, Processes (P_1, P_2, P_3)

$R =$

$P_i \rightarrow 2(R)$

a) what is $\underline{\text{Max}(R)}$ to cause deadlock 3

b) what is min (R) to ensure " freedom ? 4

$P_1 - 1$

$P_2 - 1$ 1

$P_3 - 1$

8) n - Processes

$R = 6$ Copies

$P_i \longrightarrow 3(R)$

a) What is Min(n) to cause deadlock ? 3

b) " " Max(n) for deadlock freedom ? 2

$P_1 - 2$

$P_2 - 2$

~~$P_3 - 2$~~

deadlock.

Q4)

	Max Demand R	
P_1	10	9
P_2	8	7
P_3	5	4
P_4	12	11
P_5	6	5
		36

$n = 5$

R

Deadlock

$\langle 1 \ldots 36 \rangle$

a) Max(R) to Cause deadlock : 36

b) Min(R) for deadlock freedom : 37

Q.1 Which of the following is NOT a valid Deadlock Prevention Scheme?

A. Release all resources before requesting a new resource.

B. Number all resources uniquely and never request a lower numbered resource than the last one requested.

C. Never request a resource after releasing any resource

D. Request and be allocated all required resources before execution.

Q.2 Which of the following is NOT true of deadlock prevention and deadlock avoidance schemes?

A. In deadlock prevention, the request for resources is always granted if the resulting state is safe. F

B. In deadlock avoidance, the request for resources is always granted if the resulting state is safe. T

C. Deadlock avoidance is less restrictive than deadlock prevention. T

D. Deadlock avoidance requires knowledge of resource requirements a priori. T

Q.3 An operating system implements a policy that requires a process to release all resources before making a request for another resource. Select the TRUE statement from the following:

Prevention

Hold & wait

A. Both starvation and deadlock can occur.

B. Starvation can occur but deadlock cannot occur.

C. Starvation cannot occur but deadlock can occur.

D. Neither starvation nor deadlock can occur.

Q.4 A Computer has six Tape Drives, with n-processes competing for them. Each Process may need two drives. What is the maximum value of 'n' for the System to be Deadlock free?

A. 6

B. 5

C. 4

D. 3

Q.5 An Operating System contains 3 User Processes each requiring 2 units of resource 'R'. The minimum number of units of 'R' such that no Deadlocks will ever arise is

A. 3

B. 5

C. 4

D. 6

Q.6 A Computer system has 6 Tape Drives, with 'n' Processes competing for them. Each Process may need 3 Tape Drives. The maximum value of 'n' for which the System is guaranteed to be Deadlock free is:

A. 2

B. 3

C. 4

D. 1

Q.7 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 of 3, 4 and 6 respectively. For what value of m Deadlock will not occur?

MSQ

A. 7

B. 9

C. 10

D. 13 ✓

E. 15 ✓

	Max R	
A	3 −	2
B	4 −	3
C	6 −	5
		10

$\langle 1-10 \rangle$: deadlock

>10 : " freedom

Min (R) : 11 ✓

$\langle D, E \rangle$

Q.8 Consider the following snapshot of a system running n processes. Process i is holding xi instances of a resource R, for $1 \leq i \leq n$. Currently, all instances of R are occupied. Further, for all i, process i has placed a request for an additional yi instances while holding the xi instances it already has. There are exactly two processes p and q such that $yp = yq = 0$. Which one of the following can serve as a necessary condition to guarantee that the system is not approaching a deadlock?

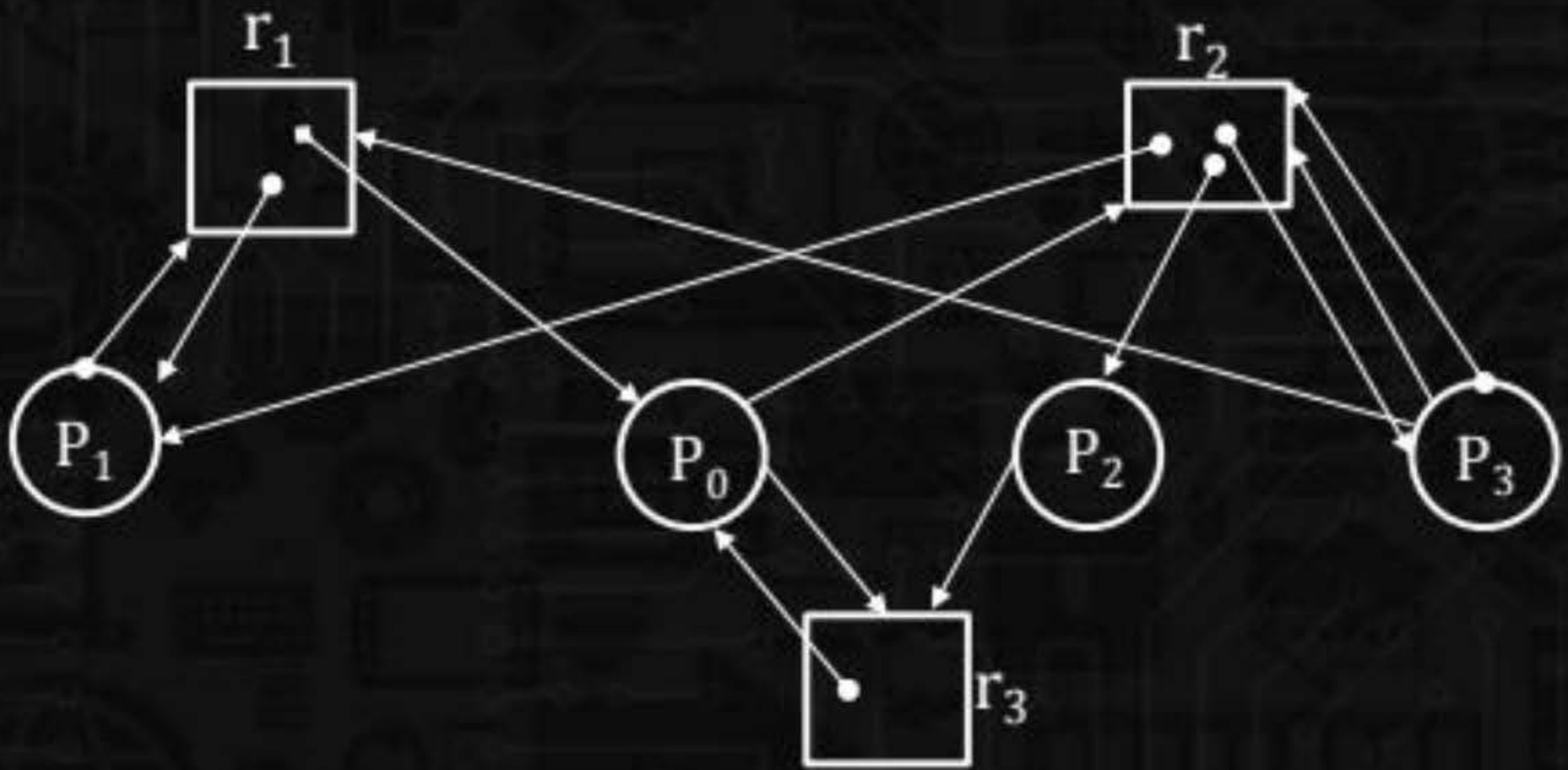
A. $\min(x_p, x_q) < \max_{k \neq p, q} y_k$

B. $x_p + x_q \geq \min_{k \neq p, q} y_k$

C. $\max(x_p, x_q) > 1$

D. $\min(x_p, x_q) > 1$

Q.9 Consider the Following Resource Allocation Graph. Find if the System is in Deadlock State.



Q.10 An operating system uses the *Banker's algorithm* for deadlock avoidance when managing the allocation of three resource types X, Y, and Z to three processes P0, P1, and P2. The table given below presents the current system state. Here, the *Allocation* matrix shows the current number of resources of each type allocated to each process and the *Max* matrix shows the maximum number of resources of each type required by each process during its execution.

	Allocation			Max		
	X	Y	Z	X	Y	Z
P0	0	0	1	8	4	3
P1	3	2	0	6	2	0
P2	2	1	1	3	3	3

There are 3 units of type X, 2 units of type Y and 2 units of type Z still available. The system is currently in a safe state. Consider the following independent requests for additional resources in the current state:

$\langle X, Y, Z \rangle = \langle 3, 2, 2 \rangle$ Avail

	Max	Alloc	Need	Avail
	X Y Z	X Y Z	X Y Z	X Y Z
P_0	8 4 3	0 0 1	8 4 2	3 2 2
P_1	6 2 0	3 2 0 5 2 0	3 0 0 1 0 0	
P_2	3 3 3	2 1 1	1 2 2	

$\langle 1, 2, 2 \rangle$

$\langle 6, 4, 2 \rangle$

$\langle 8\ 5\ 3 \rangle$

$\langle 8\ 5\ 4 \rangle$ ✓

T_0 : P_0 :$\rightarrow \langle 0, 0, 2 \rangle$ Req_1

$\langle P_1$;

Sys is unsafe

Req_1 is Not granted

T_1 : P_1 :$\rightarrow \langle 2, 0, 0 \rangle$ Req_2

$\langle P_1 ; P_2 ; P_0 \rangle$

Safe ; granted

Q.11 REQ1: P0 requests 0 units of X, 0 units of Y and 2 units of Z $\langle 0,0,2 \rangle$

REQ2: P1 requests 2 units of X, 0 units of Y and 0 units of Z

$\langle 2,0,0 \rangle$

Which one of the following is TRUE?

A. Only REQ1 can be permitted.

B. Only REQ2 can be permitted. ✓

C. Both REQ1 and REQ2 can be permitted.

D. Neither REQ1 nor REQ2 can be permitted.

Q.12 Consider a System with n Processes <P_1......P_n>. Each Process is allocated x_i copies of R (resources) and makes a request for y_i copies of R. There are exactly 2 Processes A and B whose request is zero. Further there are 'k' instances of R free available. What is the condition for stating that system is not approaching deadlock (System is said to be not approaching deadlock if minimum request of Process is satisfiable) Also compute the total instances of R in System.

Q.13 A system contains three programs, and each requires three tape units for its operation. The minimum number of tape units which the system must have such that deadlocks never arise is ________.

Q.14

A system shares 9 tape drives. The current allocation and maximum requirement of tape drives for three processes are shown below:

Which of the following best describes current state of the system?

Process	Current Allocation	Maximum Requirement
P1	3	7
P2	1	6
P3	3	5

Need	Avail
4	2
5	4
2	

A. Safe, Deadlocked

B. Safe, Not Deadlocked ✓

C. Not Safe, Deadlocked

D. Not Safe, Not Deadlocked

$\langle P_3; P_1; P_2 \rangle$

Safe

Q.15 Which of the following statements is/are TRUE with respect to deadlocks?

<A, D>

A. Circular wait is a necessary condition for the formation of deadlock. ✓

B. In a system where each resource has more than one instance, a cycle in its wait-for graph indicates the presence of a deadlock. ✗

C. ✗ If the current allocation of resources to processes leads the system to unsafe state, then deadlock will necessarily occur.

D. ✓ In the resource-allocation graph of a system, if every edge is an assignment edge, then the system is not in deadlock state.

Q.16 In a system, there are three types of resources: E, F and G. Four processes P0, P1, P2 and P3 execute concurrently. At the outset, the processes have declared their maximum resource requirements using a matrix named Max as given below. For example, Max[P2, F] is the maximum number of instances of F that P2 would require. The number of instances of the resources allocated to the various processes at any given state is given by a matrix named Allocation. Consider a state of the system with the allocation matrix as shown below, and in which 3 instances of E and 3 instances of F are the only resources available.

Allocation					Max			
	E	F	G			E	F	G
P0	1	0	1		P0	4	3	1
P1	1	1	2		P1	2	1	4
P2	1	0	3		P2	1	3	3
P3	2	0	0		P3	5	4	1

Q.17 From the perspective of deadlock avoidance, which one of the following is true?

A. The system is in safe state

B. The system is not in safe state, but would be safe if one more instance of E were available

C. The system is not in safe state, but would be safe if one more instance of F were available

D. The system is not in safe state, but would be safe if one more instance of G were available

Q.18 A multithreaded program P executes with x number of threads and uses y number of locks for ensuring mutual exclusion while operating on shared memory locations. All locks in the program are non-reentrant, i.e., if a thread holds a lock l, then it cannot re-acquire lock l without releasing it. If a thread is unable to acquire a lock, it blocks until the lock becomes available. The minimum value of x and the minimum value of y together for which execution of P can result in a deadlock are:

A. $x = 1, y = 2$

B. $x = 2, y = 1$

C. $x = 2, y = 2$

D. $x = 1, y = 1$


THANK YOU
GW
SOLDIERS !