Deadlocks

1) Consider a system which has R identical resources, 5 processes competing for them and 3 is the maximum need of each process. Then the minimum number of resources required such that deadlock will never occur is _____. **[NAT]**

Solution:

A system with R identical resources, P processes competing for them and N is the maximum need of each process, then the minimum number of resources required, so that deadlock will never occur is

$$R >= P * (N - 1) + 1$$
$$R>=5*(3-1)+1>=11.$$
Therefore, minimum =11.

2) Consider a system that has four processes and five resources. The current allocation and maximum needs are shown below:

Process	Allocated					Maximum				
P1	1	0	2	1	1	1	1	2	1	3
P2	2	0	1	1	0	2	2	2	1	0
P3	1	1	0	1	1	2	1	3	1	1
P4	1	1	1	1	0	1	1	2	2	0

If Available = [0 0 X 1 1], what is the minimum value of **x** for which this is a safe state?
A) 1
B) 2
C) 3
D) 4

Solution: (B)
Need matrix is calculated as:

Process	Need Matrix

P1	0	1	0	0	2
P2	0	2	1	0	0
P3	1	0	3	0	0
P4	0	0	1	1	0

When X = 1, Available = [0 0 1 1 1]. No process request can be granted, othe than process P4.
When X=2, Available = [0 0 2 1 1]. It can be verified that there exists a safe sequence <P4,P3,P2,P1> in which all the processes can be executed. So, the system is in a safe state when X=2.

3) Given a request from process *P* for resource *R*, a deadlock prevention algorithm is given below, where the resources have unique priorities:

Algorithm

```
if process P currently has any resources with equal or higher priority
than resources R, then
        refuse the request
else if resource R1 does not exist, then
        refuse the request
else
{
        if the resource R is not free, then
                wait until resource R is free
        end if
        grant process P exclusive access to resources R
}
```

In the above algorithm for preventing deadlock, each resource type is assigned a unique integer as a priority, with 2 as lowest priority and 0 as highest priority. Suppose

- resource R1 has priority 2,
- resource R2 has priority 1,
- resource R3 has priority 0.

Given this sequence of requests and frees:

- P1 requests R1,
- P3 requests R3,

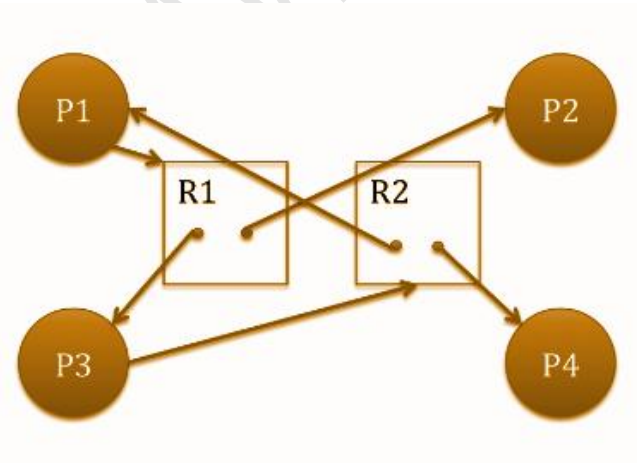
- P1 requests R2,
- P1 frees R2,
- P2 requests R2,
- P2 frees R2,
- P3 requests R2,
- P3 frees R2

Which of the following are correct? **[MSQ]**
A) P1 requests R1: P1 is granted R1
B) P3 requests R3 ==> P3 is denied R3
C) P1 requests R2 ==> P1 is granted R2 because its priority is higher than R1
D) P3 frees R2 ==> no change because its was never granted

Solution: (A)(C)(D)
P1 requests R1 ==> P1 is granted R1
P3 requests R3 ==> P3 is granted R3
P1 requests R2 ==> P1 is granted R2 because its priority is higher than R1
P1 frees R2 ==> afterwards, P1 is still holding R1
P2 requests R2 ==> P2 is granted R2
P2 frees R2 ==> afterwards, P2 is holding nothing,
P3 requests R2 ==> refused because P3 has a higher priority resource R3,
P3 frees R2 ==> no change because its was never granted

4) Consider a system with four processes P1, P2, P3, and P4, and two resources, R1, and R2, respectively.
Each resource has two instances.
-P1 allocates an instance of R2, and requests an instance of R1;
-P2 allocates an instance of R1, and doesn't need any other resource;
-P3 allocates an instance of R1 and requires an instance of R2;
-P4 allocates an instance of R2, and doesn't need any other resource.

Which one of the following is correct ?



A) No cycle is present in the corresponding Resource Allocation Graph (RAG).
B) Cycle is present but no deadlock.
C) Cycle is present and leads to deadlock.
D) None of these.

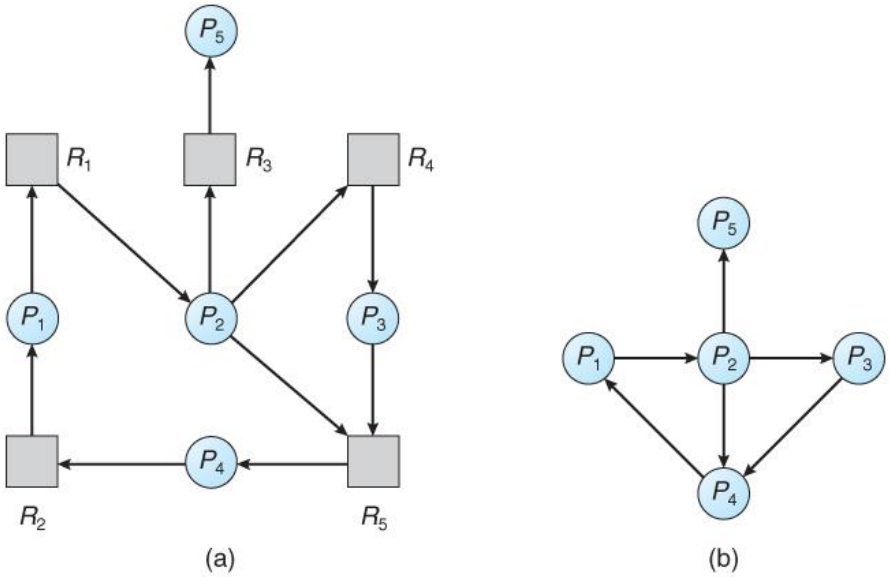
Solution: (B)
RAG:

There is a cycle, but no deadlock. P2 finishes, release R1; P4 finishes, release R2; P1 acquires R1, finishes and release R1,R2; P3 acquires R2, finishes and release R1,R2;

5) Which of the following are TRUE regarding Wait-for Graphs? **[MSQ]**

A) In a single instance resource system, cycles in the wait-for graph indicate deadlocks.
B) A wait-for graph is used for deadlock detection.
C) A wait-for graph can be constructed from a resource-allocation graph.
D) In a single instance resource system, cycles in the wait-for graph may not indicate deadlocks.

Solution: (A) (B) (C)
A wait-for graph can be constructed from a resource-allocation graph by eliminating the resources and collapsing the associated edges, as shown in the figure below.



If each resource category has a single instance, then we can use a variation of the resource-allocation graph known as a wait-for graph to detect deadlocks and, cycles in the wait-for graph indicate deadlocks

6) An operating system contains 3 user processes each requiring 2 units of resource R. The minimum number of units of R such that no deadlock will ever occur is_____
A) 1
B) 2
C) 3

D) 4

Solution: (D)

A system with R identical resources, P processes competing for them and N is the maximum need of each process, then the minimum number of resources required, so that deadlock will never occur is R >= P * (N - 1) + 1 => R >= 3*1+1 >= 4. So, the minimum is 4.

7) Which one of the following is not true?

(A) Safe state means there is no deadlock.

(B) Unsafe state always leads to a deadlock.

(C) There are four necessary conditions for a deadlock.

(D) All resources are not pre-emptable.

Solution (B)

An unsafe state may or may not lead to deadlock.

8) Which one of the following is not a deadlock recovery method?

(A) Resource pre-emption

(B) Rollback

(C) Abort the process

(D) Hold and wait

Solution (D) Hold and wait is the necessary condition of deadlock.

9) Consider the following system snapshot of four processes with 2 resources ?

Current allocation matrix

	R1	R2
P1	1	3
P2	4	1
P3	1	2
P4	2	0

Current request matrix

	R1	R2
P1	1	2
P2	4	3
P3	1	7
P4	5	1

What should be the Availability Vector such that the deadlock will not occur :

A) 1, 4

B) 2, 3

C) 2, 4

D) None of these.

Solution C)

Initial	Step 1	Step 2	Step 3	Step 4
Current allocation matrix P1 1 3 P2 4 1 P3 1 2 P4 2 0 Current request matrix P1 1 2 P2 4 3 P3 1 7 P4 5 1 Availability Vector 2 4	Give 1 2 to P1 Current allocation matrix P1 2 5 P2 4 1 P3 1 2 P4 2 0 Current request matrix P1 0 0 P2 4 3 P3 1 7 P4 5 1 Availability Vector 1 2	P1 ends, releasing 2 5 Current allocation matrix P2 4 1 P3 1 2 P4 2 0 Current request matrix P2 4 3 P3 1 7 P4 5 1 Availability Vector 3 7	Run P3 (releases 1 2): Current allocation matrix P2 4 1 P4 2 0 Current request matrix P2 4 3 P4 5 1 Availability Vector 4 9	Run P2 (releases 4 1): Current allocation matrix P4 2 0 Current request matrix P4 5 1 Availability Vector 8 10 Then P4 can run. System isn deadlocked.

10) Consider **p** processes each needing a maximum of **m** resources and a total of **r** resources available. What condition must hold to make the system deadlock free?

A) $r >= p(m - 1) + 1$

B) $r >= p(m - 1)$

C) $r <= p(m - 1) + 1$

D) $r < p(m - 1) + 1$

Solution (A)

If a process has m resources it can finish and cannot be involved in a deadlock.

Therefore, the worst case is where every process has m – 1 resources and needs another one. If there is one resource left over, one process can finish and release all its resources, letting the rest finish too. Therefore the condition for avoiding deadlock is r >= p(m – 1) + 1.

11) A computer system has 6 tape drives, with 'n' processes competing for them. Each process may need 3 tape drives. The value of 'n' for which the system is guaranteed to be deadlock free is? **[MSQ]**
a) 2
b) 3
c) 4
d) 1

Solution (A) (D)
For deadlock free operation, 3*n < 6+n, therefore, n<3, or n=1,2.

12) If 'm' processes share 'n' resources of the same type, the maximum need of each process does not exceed 'n' and the sum of all their maximum needs is always less than 'm+n'. In this case:
(a) Deadlock can never occur
(b) Deadlock may occur
(c) Deadlock has to occur
(d) None

Solution: Option (a)
The condition given implies that each process can request up to 'n' resources, and the sum of all maximum needs is less than 'm + n'. This means there are enough resources to satisfy the maximum needs of each process without exhausting all resources in the system.

13) A system is having 10 user processes each requiring 3 units of resource R. The minimum number of units of R such that no deadlock will occur _____?

Solution: The number of units that each process holds = One less than its maximum demand

So,

Process P1 holds 2 units of resource R
Process P2 holds 2 units of resource R
Process P3 holds 2 units of resource R and so on.
Process P10 holds 2 units of resource R

Thus,

Maximum number of units of resource R that ensures deadlock = 10 x 2 = 20
Minimum number of units of resource R that ensures no deadlock = 20 + 1 = 21

13) A system is having 3 user processes P1, P2 and P3 where P1 requires 2 units of resource R, P2 requires 3 units of resource R, P3 requires 4 units of resource R. The minimum number

of units of R that ensures no deadlock is _____?

Solution:
The number of units that each process holds = One less than its maximum demand

So,

Process P1 holds 1 unit of resource R
Process P2 holds 2 units of resource R
Process P3 holds 3 units of resource R

Thus,
Maximum number of units of resource R that ensures deadlock = 1 + 2 + 3 = 6
Minimum number of units of resource R that ensures no deadlock = 6 + 1 = 7

14) A system is having 3 user processes P1, P2 and P3 where P1 requires 21 units of resource R, P2 requires 31 units of resource R, P3 requires 41 units of resource R. The minimum number of units of R that ensures no deadlock is _____?

Solution:

In worst case,
The number of units that each process holds = One less than its maximum demand

So,

Process P1 holds 20 units of resource R
Process P2 holds 30 units of resource R
Process P3 holds 40 units of resource R

Thus,

Maximum number of units of resource R that ensures deadlock = 20 + 30 + 40 = 90
Minimum number of units of resource R that ensures no deadlock = 90 + 1 = 91

15) If there are 6 units of resource R in the system and each process in the system requires 3 units of resource R, then how many processes can be present at maximum so that no deadlock will occur?

Solution:

In worst case,

The number of units that each process holds = One less than its maximum demand

So,

Process P1 holds 2 units of resource R
Process P2 holds 2 units of resource R
Process P3 holds 2 units of resource R

Thus,

Minimum number of processes that ensures deadlock = 3
Maximum number of processes that ensures no deadlock = 3 – 1 = 2

16) Consider a system having m resources of the same type being shared by n processes. Resources can be requested and released by processes only one at a time. The system is deadlock free if and only if-

The sum of all max needs is < m+n
The sum of all max needs is > m+n
Both of above
None of these

Solution: (A)

Maximum number of units of resource R that ensures deadlock = ($\sum xi - n$)
Thus, For no deadlock occurrence,
Number of units of resource R must be > ($\sum xi - n$)
i.e. $m > (\sum xi - n)$
or $\sum xi < m + n$

Thus, Correct Option is (A).