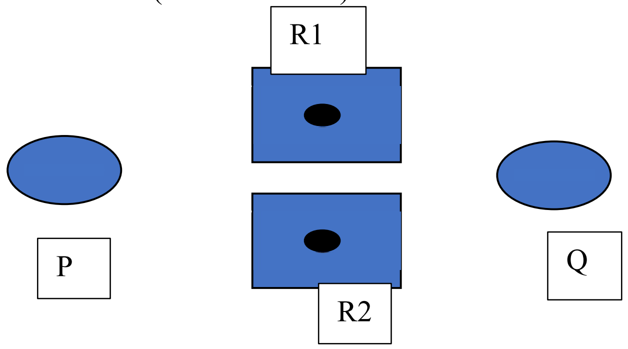
TUTORIAL-6

Q 1. Explain whether the deadlock will occur for this situation? Semaphores *A* and *B*, initialized to 1

P_0	P_1
wait (A)	*wait(B)*
wait (B)	*wait(A)*
signal(B)	*signal(A)*
signal(A)	*signal(B)*

Ans: Yes, both processes will enter deadlock at second statement.

Q 2. Prove that the process/resource configuration below cannot result in a deadlock if the Deadlock prevention method (Hold and Wait) is followed



Solution: In deadlock prevention method, either process P will acquire both R1 and R2 resources or process Q will acquire both resources at a time. None of the processes shall acquire one resource and wait for another resource which is not available.

Q 3. Consider following system(One resource class only)

Process	Holding	Max claims
A	4	6
B	4	11
C	2	7

unallocated: 2
Determine whether it is in safe state or not? If C should have a claim of 9 instead of 7, then is there any safe state?

Solution: Current Need of resources is 2, 7, 5. It is in safe state with safe sequence as A, C B.
If C should have a claim of 9 instead of 7, then Need becomes 2, 7, 7. After the execution of A, no other resource request can be granted.

Q 4. Deadlock is defined as two or more processes are waiting indefinitely for an event that can be caused by only one of the waiting processes. Give example of deadlock for 2 processes sharing some common semaphore.
Solution: Q1

Q 5. Consider the following snapshot of a system:

a. What is the content of the matrix *Need*?

b. Is the system in a safe state?

c. If a request from process *P*1 arrives for (0,4,2,0), can the request be granted immediately?

	Allocation	*Max*	*Available*
	A B C D	*A B C D*	*A B C D*
P_0	0 0 1 2	0 0 1 2	1 5 2 0
P_1	1 0 0 0	1 7 5 0	
P_2	1 3 5 4	2 3 5 6	
P_3	0 6 3 2	0 6 5 2	
P_4	0 0 1 4	0 6 5 6	

Solution:

a. The values of Need for processes P0 through P4 respectively are (0, 0, 0, 0), (0, 7, 5, 0), (1,0, 0, 2), (0, 0, 2, 0), and (0, 6, 4, 2).

b. Yes. With Available being equal to (1,5, 2, 0), either process P0 or P3 could run. Once process P3 runs, it releases its resources, which allow all other existing processes to run.

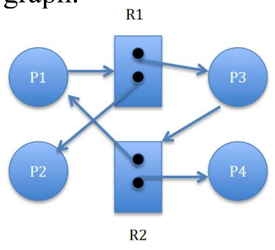
c. Yes, it can. This results in the value of Available being (1, 1, 0, 0). Need becomes (0, 0, 0, 0), (0, 3, 3, 0), (1,0, 0, 2), (0, 0, 2, 0), and (0, 6, 4, 2). One ordering of processes that can finish is P0, P2, P3, P1, and P4.

Q 6. There are four processes and two resources in the system. Each resource has two instances.

Furthermore:

- P1 acquires an instance of R2, and requests an instance of R1
- P2 acquires an instance of R1, and doesn't need any other resource
- P3 acquires an instance of R1 and requires an instance of R2
- P4 acquires an instance of R2, and doesn't need any other resource.

a) Draw the resource allocation graph.

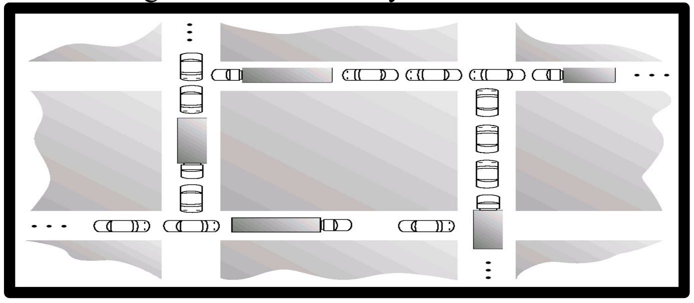


b) Is there a cycle in the graph? If yes name it.
P1 R1 P3 R2 P1

c) Whether the system is in deadlock? If yes, explain why. If not, provide sequence of execution of processes.

No. There is a cycle, but no deadlock. P2 and P4 have all resources for completing. P2 P4, P1 P3

Q 7. Show that the four necessary conditions for deadlock indeed hold in this example. State a simple rule for avoiding deadlocks in this system.



Solution: The four necessary conditions for a deadlock are (1) mutual exclusion; (2) hold-and-wait; (3) no preemption; and (4) circular wait. The mutual exclusion condition holds as only one car can occupy a space in the roadway. Hold-and- wait occurs where a car holds onto their place in the roadway while they wait to advance in the roadway. A car cannot be removed (i.e. preempted) from its position in the roadway. Lastly, there is indeed a circular wait as each car is waiting for a subsequent car to advance. The circular wait condition is also easily observed from the graphic.

Q 8. 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?

Solution: Given tape drive = 6 and each process may need 2 drive.

When we give 1 drive to 1 process then total process will be 6 but in this case there will definitely deadlock occur because every process contain 1 drive and waiting for another drive which is hold by other process therefore when we reduce 1 process then system to be deadlock free.

Hence maximum value of n = 6 – 1 = 5.

Q 9. ‘m’ processes share ‘n’ resources of the same type. The maximum need of each process doesn’t exceed ‘n’ and the sum of all their maximum needs is always less than m+n. Whether deadlock will occur or not?

Ans. No Deadlock

Q 10. Consider a system with 3 processes that share 4 instances of the same resource type. Each process can request a maximum of K instances. Resource instances can be requested and released only one at a time. What will be the largest value of K that will always avoid deadlock?

Solution: Given,
Number of processes (P) = 3
Number of resources (R) = 4

Since deadlock-free condition is: $R \geq P(N - 1) + 1$

Where R is total number of resources,
P is the number of processes, and
N is the max need for each resource.

$$4 \geq 3(N - 1) + 1$$
$$3 \geq 3(N - 1)$$
$$1 \geq (N - 1)$$
$$N \leq 2$$

Therefore, the largest value of K that will always avoid deadlock is 2.