Q.1) Max Marks: 1

Consider a system having 'm' resources of the same type. The resources are shared by 3 processes A, B, C, which have peak time demands of 3, 4, 6 respectively. The minimum value of 'm' that ensures that deadlock will never occur is

A **11** Correct Option

Solution: (A)
Answer: I
Solution: If we have m resources of same type and they are shared by 3 processes then
A needs 3 resources of same type .
B needs 4 resources of same type .
C needs 6 resources of same type .
first of all we find out after how many resource's if we give them the system still be in Deadlock.
So A needs 3 resource's of same type .If we only give 2 resource to A then it will still be in deadlock
B needs 4 resource's of same type .If we only give 3 resource to B then it will still be in deadlock
C needs 6 resource's of same type .If we only give 5 resource to C then it will still be in deadlock
we have given total of 10 resources still the system is in deadlock .To break this deadlock we give them one more.
So Answer :A (11 resources of same type).

B **12**

C **13**

D **14**

Q.2) Max Marks: 1

A solution to the Dining Philosophers Problem which avoids deadlock is:

A **Ensure that all philosophers pick up the left fork before the right fork**

B **Ensure that all philosophers pick up the right fork before the left fork**

C **Ensure that one particular philosopher picks up the left fork before the right fork, and that all other philosophers pick up the right fork before the left fork** Correct Option

Solution: (C)
Answer:C
Explanation:
according to condition, out of all, one philosopher will get both the forks. So, deadlock should not be there.

D **None of the above**

Q.3) Max Marks: 1

A thread is a lightweight process.
In the above statement, weight refers to

A **Time**

B **Number of resources** Correct Option

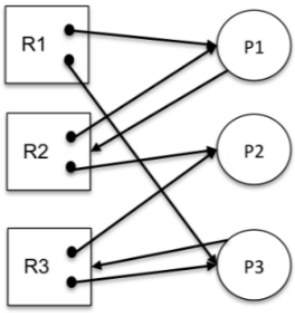
Solution: (B)
Answer:B
Explanation:
A thread is a lightweight Process.
We might call it that way because a thread, like a process, is a way to have a parallel, concurrent, flow of execution. But contrary to a process, a thread shares the same memory as the other threads in the same process, instead of having a completely separate memory.
Hence,Option(B) Number of resources is the correct choice.

C **Speed**

D **All of the above**

Q.4) Max Marks: 1

Consider the following resource allocation graph.



1: Above allocation graph contain a deadlock.

2: Assume now that P2 also demands resource R1 and in addition to that assume that R2 has now three instances then this allocation graph contains a deadlock.

Which of these statements are TRUE?

A Only 1

B Only 2

C Both 1 and 2

D None of them — Correct Option

Solution: (D)
Answer: IV
Solution:
1. No. A possible execution sequence is P2, P1, P3.
2. No. The only possible execution sequence is P1, P2, P3.

Q.5) Max Marks: 1

Consider a system having 9 resources, with n processes competing for them. Each process need 3 resources. What is the maximum value of n for the system to be deadlock free _____

Correct Answer

Solution: (4)
Answer:4
Solution:
Given, total number of resources = 9
Each process needs 3 resources
If we take 4 processes, we can allocate 2 resources to each of the process and give one extra resource to any of the process. So that it gets completed and give away its resources to their process.
With one extra process with above scenario there is a possibility of the deadlock.
∴ max number of processes is 4 so that we can guarantee deadlock free.

Q.6) Max Marks: 1

Consider the following 3 concurrent processes and resource requirements.

Process	Need
P_0	R_1, R_3
P_1	R_2, R_3
P_2	R_1, R_2

Which of the following order cause deadlock?

A $P_0 \rightarrow P_2 \rightarrow P_1 \rightarrow P_0$

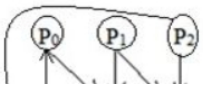
B $P_0 \rightarrow P_1 \rightarrow P_2 \rightarrow P_0$

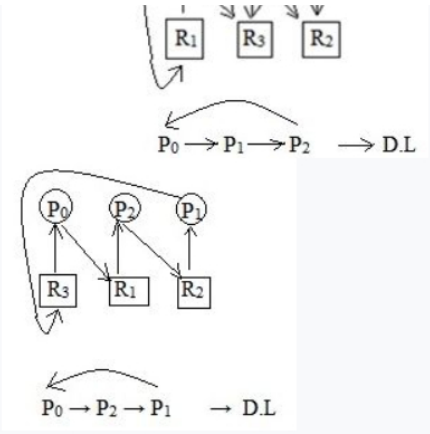
C Both $P_0 \rightarrow P_2 \rightarrow P_1 \rightarrow P_0$ & $P_0 \rightarrow P_1 \rightarrow P_2 \rightarrow P_0$ — Correct Option

Solution: (C)

Answer: III

Solution:





D None of these

Q.7) Max Marks: 1

Thread synchronization is required because

A All threads of a process share the same address space

B All threads of a process share the same global variables

C All threads of a process can share the same files

D All of the mentioned above — Correct Option

Solution: (D)
Answer:D
Explanation:
When we start two or more threads within a program, there may be a situation when multiple threads try to access the same resource and finally they can produce unforeseen results due to concurrency issues. For example, if multiple threads try to write within a same file then they may corrupt the data because one of the threads can override data or while one thread is opening the same file at the same time another thread might be closing the same file.
So there is a need to synchronize the action of multiple threads and make sure that only one thread can access the resource at a given point in time.

Q.8) Max Marks: 1

The wait-for graph is a deadlock detection algorithm that is applicable when :

A All resources have multiple instances

B All resources have a single 7 multiple instances

C All resources have a single instance — Correct Option

Solution: (C)
Answer: C
Explanation:
The wait-for-graph scheme is not applicable to a resource allocation system with multiple instances of each resource type.

D All of the mentioned above

Q.9) Max Marks: 1

Consider the following statements

a. The only way a thread can modify another thread's stack variable is if the thread is passed the address of that variable as an argument during creation.
b. When a hardware interrupt occurs, the kernel enqueues the interrupt and serves it when the current running thread completes

Which of these are not TRUE?

A Only a

B Only b

C Both a & b — Correct Option

Solution: (C)
Answer: III
Solution:

a. Threads in the same address space can access any memory address (example: a global pointer to a local stack variable)
b. On an interrupt the current running thread is switched out and the interrupt serviced.

D None of these

Q.10) Max Marks: 1

With single resource, deadlock occurs

A if there are more than two processes competing for that resources

B if there are only two processes competing for that resources

C if there is a single process competing for that resources

D None of these Correct Option

Solution: (D)
Soln) D
Is it processor or process any way assuming it is process then surerly C can not be the ans
option B 2 process are competing for the resource one can be given and let executed so no deadlock
option A more than 2 processes competing again any one can be allocated and hence execute since there is no circular wait
ans should be D none of these

Q.11) Max Marks: 2

Consider the following statements

a. Switching a thread is costlier than switching a process.
b. Suppose a user level thread is running in a critical section of code, meaning that it has acquired all the locks through proper arbitration. Even though it can get context switched.
c. A thread can hold only one lock at a time.

Which of the following are not TRUE?

A Only a

B Only a & c Correct Option

Solution: (B)
Answer: II
Solution:
a. FALSE, Less state needs to be saved and restored. Furthermore, switching between threads benefits from caching; whereas, switching between processes invalidates the cache and TLB
b. TRUE, a process holding a lock can get context switched. Locks (especially user-level locks) are independent of the scheduler. (Note that threads running in the kernel with interrupts disabled would not get context-switched).
c. FALSE, a thread can hold/acquire any number of locks simultaneously

C a, b & c

D Only b

Q.12) Max Marks: 2

Consider the following statements

a. Threads that are part of the same process share the same general-purpose registers
b. With kernel-level threads, if one thread of a process blocks, all the threads of that process will also be blocked
c. Deadlock can be avoided by using semaphores instead of locks for mutual exclusion.

Which of the following statements are FALSE?

A Only a

B Only a & b

C Only b & c

D All of these Correct Option

Solution: (D)
Answer: IV
Solution:
- False; each thread must have own registers (saved and restored

a. False; each thread must have own registers (saved and restored when that thread is not executing).
b. False; with kernel-level threads, the OS is aware of the state of each thread and just that thread will be blocked
c. False; if you use a semaphore for mutual exclusion, it has all the same properties as a traditional lock.

Q.13) Max Marks: 2

Consider the following statements

a. Cyclic dependency always lead to deadlock.
b. There is a chance that the deadlock can happen though we follow bankers algorithm for allocating resources.
c. Message passing is faster than shared resource for communication between two processes.

Which of the following is/are TRUE?

A Only c

B Only a and b

C Only b and c

D None of these — Correct Option

Solution: (D)

Answer: IV

Solution:

a. FALSE, If multiple equivalent resources exist, then a cycle could exist that is not a deadlock. The reason is that some thread that is not part of the cycle could release a resource needed by a thread in the cycle, thereby breaking the cycle.
b. FALSE, Deadlock avoidance ensures the system is always in a safe state by not granting requests that may move the system to an unsafe state. A is considered safe if it is possible for all processes to finish executing (i.e. a sequence exists such that each process can be given all its required resources, run to completion, and return resources allocated, thus allowing another process to do the same, etc until all process complete). Deadlock avoidance requires the system to keep track of the resources such that it knows the allocated, available, and remaining resource needs. The Bankers algorithm is a deadlock avoidance scheme since it defines an algorithm and structure to ensure the system remains in a safe state before granting any new resource requests.
c. False, Communicating via shared memory is often faster and more efficient since there is no kernel intervention and no copying.

Q.14) Max Marks: 2

Suppose that we have the following resources: A, B, C and threads T1, T2, T3, T4. The total number of each resource is:

Total		
A	B	C
12	9	12

Further, assume that the processes have the following maximum requirements and current allocations:

Thread ID	Current Allocation			Maximum		
	A	B	C	A	B	C
T1	2	1	3	4	9	4
T2	1	2	3	5	3	3
T3	5	4	3	6	4	3
T4	2	1	2	4	8	2

Which of the following is a most suitable answer?

A The system is in safe state & has only one possible sequence to complete all of the tasks. — Correct Option

Solution: (A)

The system is in Safe state and has more than one possible completion sequence.

To prove this, we first compute the currently free allocations:

Available		
A	B	C
2	1	1

Further, we compute the number needed by each thread (Maximum – Current Allocation):

Thread ID	Needed Allocation		
	A	B	C
T1	2	8	1
T2	4	1	0
T3	1	0	0
T4	2	7	0

Thus, we can see that a possible sequence is: T3, T2, T4, T1:

Thread ID	Needed Allocation			Current Allocation			Available Before		
	A	B	C	A	B	C	A	B	C
T3	1	0	0	5	4	3	2	1	1
T2	4	1	0	1	2	3	7	5	4
T4	2	7	0	2	1	2	8	7	7
T1	2	8	1	2	1	3	10	8	9

Hence, it is safe state and have only possible sequence.

B The system is in unsafe state.

C The system is in safe state & has more than one possible sequence to complete all of the tasks.

D Nothing can be said about this.

Q.15) Max Marks: 2

Consider a system having "n" resources of the same type. These resources are shared by 3 processes, A, B, C. These have peak demands of 3, 4, and 6 respectively. For what value of "n" deadlock won't occur.

A 15

B 9

C 10

D 13 Correct Option

Solution: (D)
Answer: IV
Solution:
Apply the PigeonHole principle.
value of n will be (3-1) + (4-1) + (6-1) + 1= 11
but here there is no option which is 11.
now lets try 2nd way.
it may be possible that after completion of task processes don't want to release resources (Yes it is not written anywhere that processes must release resources after Execution) so we can think like this also.
To overcome this situation we have to allocate all required resources to each process at very stating. i.e. 3+4+6 = 13
13 satisfy both cases. so in the worst case it should be the answer.

close