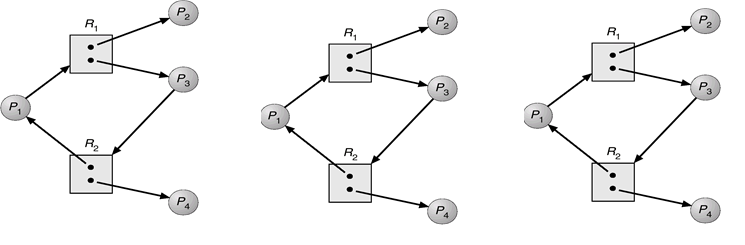


ITSC205: Operating Systems Internals

1. Which of the following resource-allocation graph with 4 processes P1,P2,P3,P4 and two instances of each resource (R1 and R2) is in deadlock?



1. B) C)
2. A deadlocked state occurs whenever \_\_\_\_.
3. a process is waiting for I/O to a device that does not exist
4. the system has no available free resources
5. A process in a set is waiting for a resource(event) that can only be released by another process that is holding it in the set
6. a process is unable to release its request for a resource after use
7. One necessary condition for deadlock is \_\_\_\_, which states that at least one resource must be held in a nonsharable mode.
8. hold and wait
9. mutual exclusion
10. circular wait
11. no preemption
12. One necessary condition for deadlock is \_\_\_\_\_\_, which states that a process must be holding one resource and waiting to acquire additional resources.
13. hold and wait
14. mutual exclusion
15. circular wait
16. no preemption
17. One necessary condition for deadlock is \_\_\_\_\_\_, which states that a resource can be released only voluntarily by the process holding the resource.
18. hold and wait
19. mutual exclusion
20. circular wait
21. no preemption
22. One necessary condition for deadlock is \_\_\_\_\_\_, which states that there is a chain of waiting processes whereby P0 is waiting for a resource held by P1, P1 is waiting for a resource held by P2, and Pn is waiting for a resource held by P0.
23. Atomic operations Test-and-Set
24. mutual exclusion
25. circular wait
26. no preemption
27. To handle deadlocks, operating systems most often \_\_\_\_\_.
28. pretend that deadlocks never occur
29. use protocols to prevent or avoid deadlocks
30. detect and recover from deadlocks
31. None of the above
32. Suppose that there are ten resources available to three processes. At time 0, the following data is collected. The table indicates the process, the maximum number of resources needed by the process, and the number of resources currently owned by each process. Which of the following correctly characterizes this state?

|  |  |  |
| --- | --- | --- |
| Process | Maximum Needs | Currently Owned |
| P0 | 10 | 4 |
| P1 | 3 | 1 |
| P2 | 6 | 4 |

1. It is safe.
2. It is not safe.
3. The state cannot be determined.
4. It is an impossible state.
5. A race condition \_\_\_\_.
6. results when several threads try to access the same data concurrently
7. results when several threads try to access and modify the same data concurrently
8. will result only if the outcome of execution does not depend on the order in which instructions are executed
9. None of the above
10. An instruction that executes atomically \_\_\_\_.
11. must consist of only one machine instruction
12. executes as a single, uninterruptible unit
13. cannot be used to solve the critical section problem
14. All of the above
15. A semaphore \_\_\_\_.
16. is essentially an integer variable
17. is accessed through only one standard operation
18. can be modified simultaneously by multiple threads
19. cannot be used to control access to a thread's critical sections
20. Which of the following synchronization techniques implement Wait( ) and Signal( ) atomic operations?
21. Mutex Locks
22. Semaphores
23. Spinlock
24. Hardware instructions
25. A spinlock \_\_\_\_.
26. is never advantageous
27. will ultimately result in a context switch when a process must wait on a lock
28. does not require a context switch when a process must wait on a lock
29. is useful when locks are expected to be held for long amounts of time
30. A(n) \_\_\_\_\_\_\_ refers to where a process is accessing/updating shared data.
31. critical section
32. entry section
33. mutex
34. test-and-set
35. When using semaphores, a process invokes the wait() operation before accessing its critical section, followed by the signal() operation upon completion of its critical section. Consider reversing the order of these two operations—first calling signal(), then calling wait(). What would be a possible outcome of this?
36. Starvation is possible.
37. Several processes could be active in their critical sections at the same time.
38. Mutual exclusion is still assured.
39. Deadlock is possible.
40. A \_\_\_\_\_ could be preempted from a process.
41. mutex lock
42. CPU
43. Semaphore
44. file lock
45. What is the most common synchronization technique implemented by Linux and Windows operating systems for Simultaneous Processing systems? Explain the technique