ⓘ Students have either already taken or started taking this quiz, so take care when editing it. If you change any quiz questions in a significant way, you might want to consider re-grading students' quizzes who took the old version of the quiz.

Points 10 Published ⋮

Details | Questions

Show question details

## Question 1 pts

Given the following code which shows how to use atomic increment operation using `CompareAndSwap` atomic operation:

```
1  void AtomicIncrement(int *value, int amount) {
2    do {
3      int old = *value;
4    } while (CompareAndSwap(value, old, old + amount) == 0);
5  }
```

Assuming that we use this `AtomicIncrement` operation to increment a shared variable, will this code ever deadlock?

- Yes it will deadlock
- No it will never deadlock
- May be it does some times?
- None of the options

## Question 1 pts

If any of the following conditions is not met, the deadlock cannot occur?

- Mutual Exclusion
- Hold-and-Wait
- Circular Wait
- No Preemption

- True
- False

## Question 1 pts

What can be used when acquiring locks to prevent deadlock?

- ordering
- consistency
- locking
- serialization

## Question 1 pts

The following code is an example of _____ which helps prevent deadlock?

```
pthread_mutex_lock(p);
pthread_mutex_lock(L1);
pthread_mutex_lock(L2);
...
pthread_mutex_unlock(p);
```

- hold-and-wait
- circular wait
- atomicity
- none of the options

## Question 1 pts

instead of lock, trylock function may better help in preventing deadlock

- True
- False

## Question 1 pts

_____ condition will not happen in concurrent system if we are implementing deadlock avoidance technique?

- circular wait
- hold-and-wait
- atomicity
- synchronization

## Question 1 pts

____ technique is used to achieve concurrency within a program without using multi-threading

- event loop
- fork/wait
- POSIX Threading
- std::thread

## Question 1 pts

You are given the following contention table:

| | T1 | T2 | T3 | T4 |
|---|---|---|---|---|
| L1 | yes | yes | yes | no |
| L2 | yes | yes | yes | no |

Which of the following schedules is guaranteed to never deadlock on a dual core CPU?

iswer

- T1 on Core 1 and T4 on Core 2
- T2 on Core 1 and T1 on Core 2
- T3 on Core 1 and T2 on Core 2
- T2 on Core 1 and T3 on Core 2

## Question 1 pts

What is livelock?

iswer

- Two threads are both repeatedly attempting to acquire lock L1 and then trylock on L2 but repeatedly failing to acquire both locks.
- One thread repeatedly attempting to acquire lock L1 and another thread T2 tries to acquire the lock L2 but both fail repeatedly to acquire their locks.
- None of the threads T1 or T2 acquires the lock L1
- The first thread acquires the lock L1 and the second thread waits indefinitely.

## Question 1 pts

We can use the `compareAndSwap` atomic operation to implement a lock-free data structure?

iswer

- True
- False

+ New question + New question group Find questions

Notify users this quiz has changed

Cancel Save