

My Courses All Courses About Contact



Help

vitorpbarbosa7



Course

**Progress** 

Course > Readings > Reading 9: Locks & Synchronization > Questions



# Questions

#### □ Bookmark this page

## Deadlock

4/4 points (graded)

In the code below three threads 1, 2, and 3 are trying to acquire locks on objects alpha, beta, and gamma.

## Thread 1

```
synchronized (alpha) {
    // using \alpha
    // ...
synchronized (gamma) {
    synchronized (beta) {
         // using \beta & \gamma
// finished
```

#### Thread 2

```
synchronized (gamma) {
    synchronized (alpha) {
         synchronized (beta) {
              // using \alpha, \beta & \gamma
              // ...
// finished
```

## Thread 3

```
synchronized (gamma) {
    synchronized (alpha) {
         // using \alpha & \gamma
         // ...
}
synchronized (beta) {
    synchronized (gamma) {
         // using \beta & \gamma
         // ...
// finished
```

This system is susceptible to deadlock.

For each of the scenarios below, determine whether the system is in deadlock if the threads are currently on the indicated lines of code.

#### Scenario A

```
Thread 1 inside using alpha
Thread 2 blocked on synchronized (alpha)
Thread 3 finished
```

## Page 2

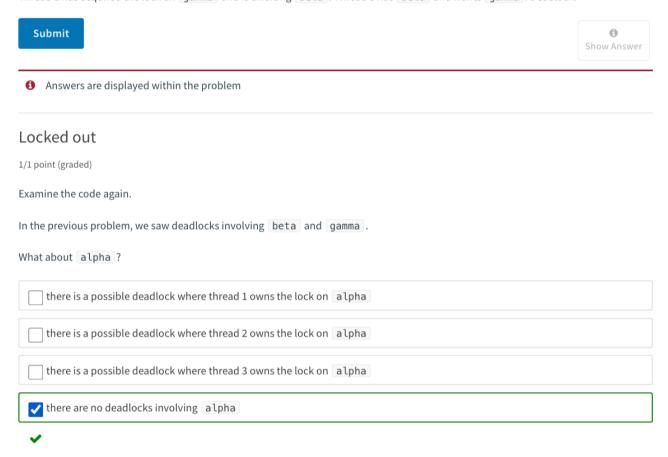
Questions | Reading 9: Locks & Synchronization | 6.005.2x Courseware | MIT Open Learning Library https://openlearninglibrary.mit.edu/courses/course-v1:MITx+6.005.2x+1T2017/courseware/Readings/09-Locks-Synchronization/?child=first

deadlock
not deadlock
<b>✓</b>
<b>Explanation</b> Thread 1 will exit the top synchronized block, release the lock on alpha, and the system will continue.
Scenario B
Thread 1 finished Thread 2 blocked on synchronized (beta) Thread 3 blocked on 2nd synchronized (gamma)
deadlock
onot deadlock
•
Explanation Thread 2 has acquired the lock on gamma and is awaiting beta . Thread 3 has beta and wants gamma . Deadlock.
Scenario C
Thread 1 running synchronized (beta) Thread 2 blocked on synchronized (gamma) Thread 3 blocked on 1st synchronized (gamma)
deadlock
not deadlock
<b>✓</b>
Explanation Thread 1 can successfully acquire the lock on beta, then exit the synchronized block, and one of the other threads will be able to acquire the lock on gamma. (As we saw in scenario B, they could deadlock later!)
Scenario D
Thread 1 blocked on synchronized (beta) Thread 2 finished Thread 3 blocked on 2nd synchronized (gamma)
deadlock
onot deadlock

https://openlearninglibrary.mit.edu/courses/course-v1:MITx+6.005.2x+1T2017/courseware/Readings/09-Locks-Synchronization/?child=first-fir

#### Explanation

Thread 1 has acquired the lock on gamma and is awaiting beta. Thread 3 has beta and wants gamma. Deadlock.



## **Explanation**

We can reason about it this way: in order to encounter deadlock, threads must try to acquire locks in different orders, creating a cycle in the graph of who-is-waiting-for-who.

So we look at alpha vs. beta: are there two threads that try to acquire these locks in the opposite order? No. Only thread 2 acquires them both at the same time.

Next we look at alpha vs. gamma: are there two threads that try to acquire these locks in the opposite order? No. Both thread 2 and thread 3 acquire both locks, but both of them acquire gamma first, then alpha.

Use the same analysis to demonstrate why beta and gamma are susceptible to deadlock: what order are these locks acquired in by the different threads?



6 Answers are displayed within the problem



 $\label{thm:local_Questions} Questions \ | \ Reading \ 9: Locks \ \& \ Synchronization \ | \ 6.005.2x \ Courseware \ | \ MIT \ Open \ Learning \ Library \ https://openlearninglibrary.mit.edu/courses/course-v1: MITx+6.005.2x+1T2017/courseware/Readings/09-Locks-Synchronization/?child=first \ https://openlearninglibrary.mit.edu/courses/course-v1: MITx+6.005.2x+1T2017/course-ware/Readings/09-Locks-Synchronization/?child=first \ https://openlearninglibrary.mit.edu/course-ware/Readings/09-Locks-Synchronization/?child=first \ https://openlearnings/09-Locks-Synchronization/?child=first \ https://openlearnings/09-Locks-Synchronization/?child=first \ https://openlearnings/09-Locks-Synchronization/?child=first \ https://openlearnings/09-Locks-Synchronization/?child=first \ https://openlearnings/09-Locks-Synchronization/?child=first \ https://openlearnings/09-Locks-Synchronization/propenlearnings/09-Locks-Synchronization/propenlearnings/09-Locks-Synchronization/propenlearnings/09-Locks-Synchronization/propenlearnings/09-Locks-Synchronization/propenlearnings/09-Locks-Synchronization/propenl$ 

△ ③ Some Rights Reserved

Open Learning Library Connect

About Contact

Accessibility Twitter

All Courses Facebook

Why Support MIT Open Learning?

Help

Privacy Policy Terms of Service

© Massachusetts Institute of Technology, 2025