CSCD 467/567 Lab 7 Avoid Deadlock Using Lock Hierarchy and tryLock

Submission

Please submit your java source file and put all files into a zip file and upload it on EWU Canvas.

Problem Description

Your program has to implement the following features.

1. Based upon the source code provided in the folder named as LabDinerFixDeadLock, you have to modify the provided code to fulfill the following.
2. You have to use the Lock class from concurrent.locks package, and use lock.tryLock() method.
3. Number each fork from 0 to 5. Philosopher always tries to grab the lower-numbered fork. Then tries to grab the high-numbered fork.
4. **After the lower-numbered fork has been acquired,**
    a. Philosopher tries to grab the high-numbered fork using tryLock().
    b. If lock is available, the philosopher grab it and eat.
    c. If the lock(the second fork) is NOT available, we increment a counter called numTry by 1, Then pause the philosopher for a while using time wait method wait(ms).
    d. If numTry reaches the allowedMaxTry = 3, then the philosopher put down what he had acquired (the lower-numbered fork).
    e. After voluntarily put down the first forks, it should notify other thread that is in timing wait.
    f. You can refer to the code here for your convenience.

    http://docs.oracle.com/javase/tutorial/essential/concurrency/newlocks.html