Extend the transferFunds() method from Lecture 26 (shown below) to use j.u.c. locks with tryLock() instead of synchronized, and to return a boolean value --- true if it succeeds in obtaining both locks and performing the transfer, and false otherwise. Assume that each Account object contains a reference to a dedicated ReentrantLock object. Sketch your answer below using pseudocode. Can you create a deadlock with multiple calls to transferFunds() in parallel?

1. `public void transferFunds(Account from, Account to, int amount) {`
2.     `synchronized (from) {`
3.         `synchronized (to) {`
4.             `from.subtractFromBalance(amount);`
5.             `to.addToBalance(amount);`
6.         }
7.     }
8. }`