- Deposit (amount)
- Withdraw (amount)

When a husband and wife share a bank account. This situation when they using two function above simultaneously may occur ***race condition.*** Possible outcomes we could get:

1. Husband and wife use withdraw (amount) at the same time:
    a. It may only decrease an amount that one function call.
    b. E.g.: - withdraw (500)
       - withdraw (1000)

       It can load the balance the same time and assign the balance back by the last executing function.
2. Husband and wife use deposit (amount) at the same time:
    a. It may only increase an amount that one function call.
    b. E.g.: - Deposit (500)
       - Deposit (1000)

       It can load the balance the same time and assign the balance back by the last executing function.
3. Husband and wife use withdraw (amount) and deposit (amount) at the same time:
    a. It may only calculate wrong amount:
    b. E.g.: - Deposit (500)
       - Deposit (1000)

       It can load the balance the same time and assign the balance back by the last executing function.

The bank can use a mutex lock to lock the balance when an account try to change their balance. The mutex lock will lock all the other critical section that try to change the balance and release when the process end. This implement will avoid the race condition for this situation and more.

Hacking Banks With Race Conditions - The Startup

This show how race condition may affect if we do not care enough about this when implement the system.