DEADLOCK

1. INTRODUCTION

In this report, we analyze the problems with the given code and find solutions to them.

2. THE PROBLEM OF THE CODE AND HOW TO SOLVE IT?

In the given code, the f1 function try to acquire the l1 lock and then the l2 lock while the f2 function is trying to acquire the l2 lock and then the l1 lock. In other words, a deadlock is occurring as the two threads hold locks on each other.

```python
import threading
import time

l1=threading.Lock()
l2 = threading.Lock()

def f1(name):
    print('thread',name,'about to lock l1')
    with l1:
        print('thread',name,'has lock l1')
        time.sleep(0.3)
        print('thread',name,'about to lock l2')
        with l2:
            print('thread',name,'run into deadLock,\nthis line will never run')

def f2(name):
    print('thread',name,'about to lock l2')
    with l2:
        print('thread',name,'has lock l2')
        time.sleep(0.3)
        print('thread',name,'about to lock l1')
        with l1:
            print('thread',name,'run into deadLock,\nthis line will never run')

if __name__ == '__main__':
    t1=threading.Thread(target=f1, args=['t1',])
    t2=threading.Thread(target=f2, args=['t2',])

    t1.start()
    t2.start()
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

/opt/homebrew/bin/python3 /Users/misatoseki/ComputersPlatformlsAndOS/Session16/deadlock.py
misatoseki@MisatonoMacBook-Air Session16 % /opt/homebrew/bin/python3 /Users/misatoseki/ComputersPlatformlsAndOS/Session16/deadlock.py
thread t1 about to lock l1
thread t1 has lock l1
thread t2 about to lock l2
thread t2 has lock l2
thread t1 about to lock l2
thread t2 about to lock l1
```

Picture 1. given code and executed result

To avoid this deadlock, the acquire() method should be used to acquire the lock and the release() method should be used to release the lock. The acquire() method can also has a blocking argument. Setting the blocking argument to False allows processing to proceed without blocking if lock acquisition fails.

```python
import threading
import time

l1 = threading.Lock()
l2 = threading.Lock()

def f1(name):
    print('thread', name, 'about to lock l1')
    if l1.acquire(blocking=False):
        try:
            print('thread', name, 'has lock l1')
            time.sleep(0.3)
            print('thread', name, 'about to lock l2')
            if l2.acquire(blocking=False):
                try:
                    print('thread', name, 'has lock l2')
                    print('thread', name, 'continuing execution')
                finally:
                    l2.release()
            else:
                print('thread', name, 'failed to acquire lock l2')
        finally:
            l1.release()
    else:
        print('thread', name, 'failed to acquire lock l1')
```

```python
def f2(name):
    print('thread', name, 'about to lock l2')
    if l2.acquire(blocking=False):
        try:
            print('thread', name, 'has lock l2')
            time.sleep(0.3)
            print('thread', name, 'about to lock l1')
            if l1.acquire(blocking=False):
                try:
                    print('thread', name, 'has lock l1')
                    print('thread', name, 'continuing execution')
                finally:
                    l1.release()
            else:
                print('thread', name, 'failed to acquire lock l1')
        finally:
            l2.release()
    else:
        print('thread', name, 'failed to acquire lock l2')

if __name__ == '__main__':
    t1 = threading.Thread(target=f1, args=['t1',])
    t2 = threading.Thread(target=f2, args=['t2',])

    t1.start()
    t2.start()
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

/opt/homebrew/bin/python3 /Users/misatoseki/ComputersPlatformlsAndOS/Session16/deadlock2.py
misatoseki@MisatonoMacBook-Air Session16 % /opt/homebrew/bin/python3 /Users/misatoseki/ComputersPlatformlsAndOS/Session16/deadlock2.py
thread t1 about to lock l1
thread t1 has lock l1
thread t2 about to lock l2
thread t2 has lock l2
thread t2 about to lock l1
thread t1 about to lock l2
thread t2 failed to acquire lock l1
thread t1 failed to acquire lock l2
misatoseki@MisatonoMacBook-Air Session16 %
```

Picture 2. edited code and executed result

3. WHAT WE LEARNED

It was found that the order acquiring and releasing of locks must be properly managed by using such as acquire () method in order to avoid deadlocks.