Lab 10

Deadlock

```
class A {
    synchronized void foo(B b) {
        String name = Thread.currentThread().getName();
        System.out.println(name + " entered A.foo");
        try {
            Thread.sleep(1000)
        } catch (Exception e) {
            System.out.println("A interrupted")
        }
        System.out.println(name + " trying to call
                                    B.last()");
        b.last();
    }
    void last() {
        System.out.println(name + " inside A.last");
    }
}
class B {
    synchronized void bar(A a) {
        String name = Thread.currentThread().getName
                                    ();
        System.out.println(name + " entered B.bar");
        try {
            Thread.sleep(1000);
        } catch (Exception e) {
            System.out.println(name +
                " trying to call A.last()");
        }
        a.last();
    }
    void last() {
```

```
        System.out.println("inside A.last");
    }
}

class Deadlock implements Runnable
{
    A a = new A();
    B b = new B();
    Deadlock() {
        Thread.currentThread().setName("MainThread");
        Thread t = new Thread(this, "RacingThread");
        t.start();
        a.foo(b);
        System.out.println("back in main thread");
    }

    public void run() {
        b.bar(a);
        System.out.println("back in other thread");
    }

    public static void main(String args[]) {
        new Deadlock();
    }
}
```

Main Thread enters A.foo
RacingThread entered B.bar
MainThread trying to call B.last()
Inside A.last
RacingThread trying to call A.last()
Inside A.last
Back in other thread
Back in main thread

8
13/2/24