# LAB 10

```
class A {
    synchronized void Foo(B b){
        String name = Thread.currentThread().getName();
        SOP(name + " entered A.foo");

        try {
            Thread.sleep(1000);
        } catch(Exception e) {
            SOP("A interrupted");
        }

        SOP(name + " trying to call B.last()");
        b.last();
    }
    synchronized void last(){
        System.out.println("Inside A.last");
        }
}

class B {
    synchronized void bar(A a){
        String name = Thread.currentThread().getName();
        SOP(name + "entered B.bar");
```

```
try {
    Thread.sleep(1000);
  } catch (Exception e) {
      SOP(" B interrupted");
  }
  SOP(name + " trying to call A.last()");
  a.last();
}

synchronized void last(){
    SOP("Inside B.last");
  }
}

class Deadlock implements Runnable {
  A a = new A();
  B b = new B();

  Deadlock(){
    Thread.currentThread().setName("Main Thread");
    Thread t = new Thread(this, "Racing Thread");
    t.start();

    a.Foo(b);
    SOP("Back in main thread");
  }
```

```
public void run() {
    b.bar(a);
    sop("Back in other thread");
}

psvm(String[] args) {
    new Deadlock();
}
}
```

Output

RacingThread entered B.bar

Main Thread entered A.Foo

Main Thread trying to call B.last()

RacingThread trying to call A.last()