

1 1 \*

;

Synchronized void foo(B b)

```
get Name();
```

System out. prntln ( $\frac{1}{A}$  interrupted);

```
3 System.out.println("name + " entered  
A. Foo");
```

boy

Thread, Slip (1000);

3

catch (Exception e)

3

```

} System.out.println("A Interrupted");

```

```
System.out.println(name + " trying to  
call B.last()");
```

b. last(),

3

3

class B

{

    synchronized void bar(Aa)

    {

        String name = Thread.currentThread().getName();

        System.out.println(name + "Entered B.bar()");

        try {

            Thread.sleep(1000);

        }

        catch (Exception e)

        {

            System.out.println("B Interrupted")

        }

        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  
("Main Thread");

Thread t = new Thread(this,  
"Racing Thread");

t.start();

a.foo(b);

~~thread~~

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();

}

}

⇒ OUTPUT:

MainThread entered A.foo

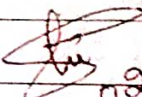
RacingThread entered B.bar

MainThread trying to call B.last()  
inside A.last

Back in main Thread

RacingThread trying to call A.last()  
inside A.last

Back in other thread

  
13.02.24