

Got: 2	Got: 4	Got: 6	Got: 8
Put: 3	Put: 5	Put: 7	Put: 9
Got: 3	Got: 5	Got: 7	Got: 9
Put: 4	Put: 6	Put: 8	Put: 10
			Got: 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("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("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);

```

```
        System.out.println("Back in main thread");

```

```
    }

```

Output:-

Main Thread entered A.foo

Racing Thread entered B.bar

Main Thread trying to call B.last()

Inside A.last.

Back in main thread

Racing Thread trying to call A.last()

Inside A.last

Back in other thread