```
class A
{
 synchronized void foo(B b)
  {
    String name = Thread.currentThread().getName();
     System.out.print(name+" entered A.foo");
     try{ Thread.sleep(1000); }
    catch (Exception e)
      {
       System.out.println("A Interrupted");
      }
     System.out.print(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();
     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();
       }
     synchronized 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();
    t.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();
         }
           DivisionMain dm = new DivisionMain();
            dm.setSize(new Dimension(800,400));
            dm.setTitle("Division of Integers");
             dm.setVisible(true);
           }
        }
```

Output

```
mainThread entered A.foo.
RacingThread Execution Started B.bar.
mainThread trying to call B.last();
```