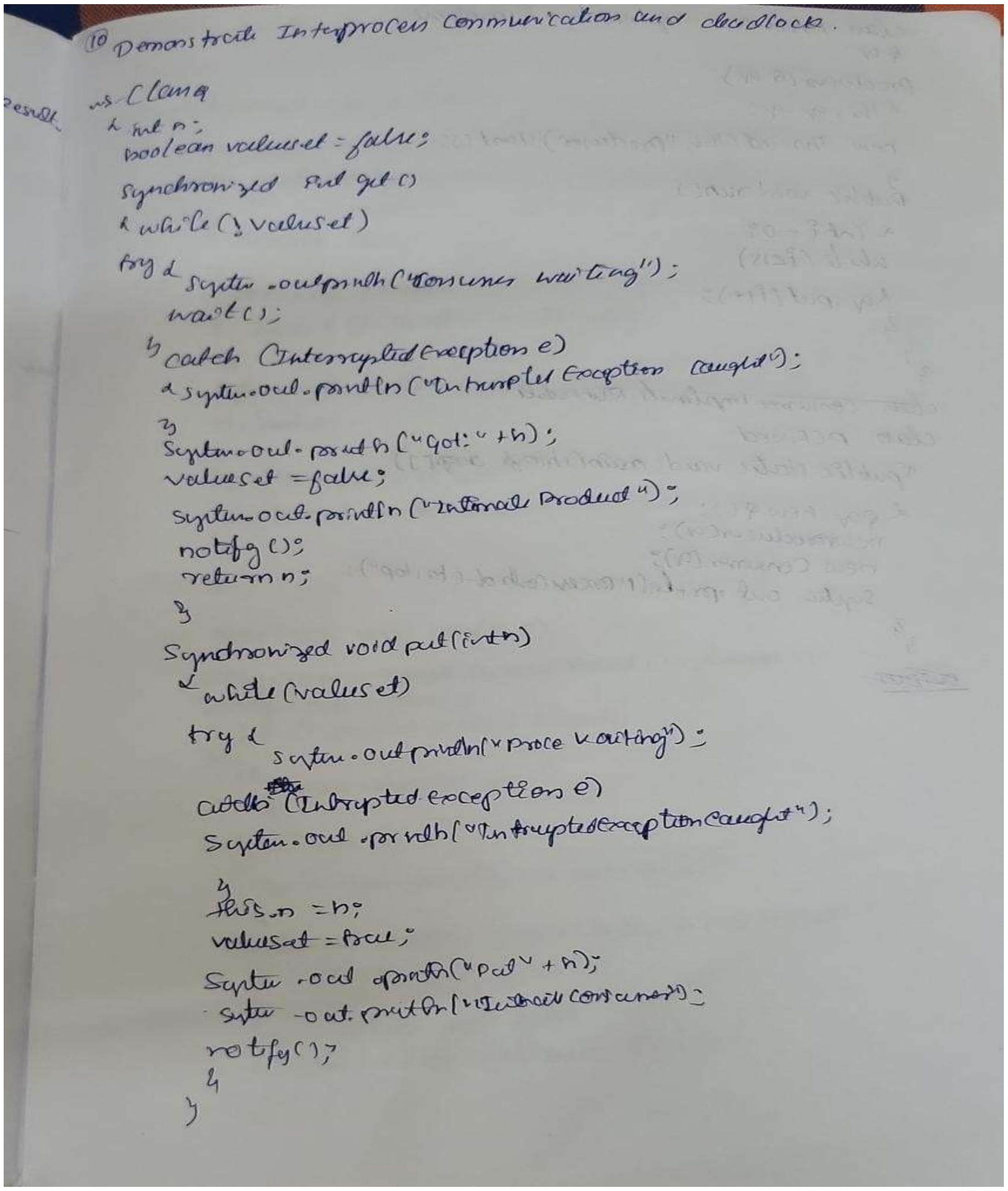
PROGRAM 10 :

Demonstrate Inter process Communication and deadlock

1. Demonstration of Inter process Communication Observation

OBSERVATION:



⑩ Demonstrate Interprocess Communication and deadlock.

Result

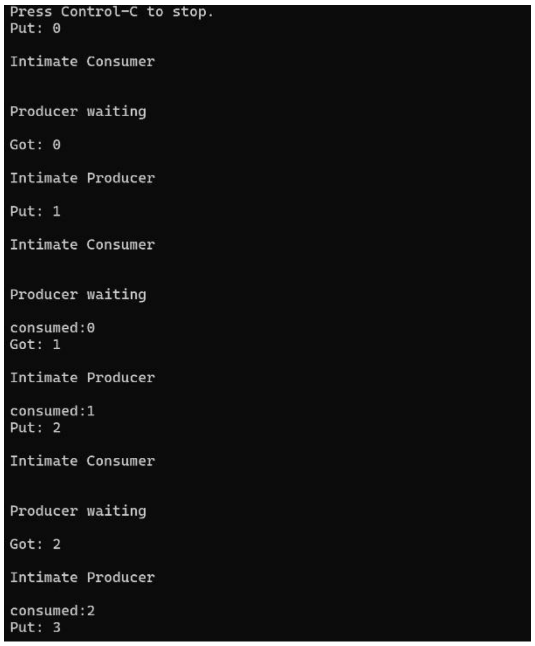
```
class Q
{ int n;
boolean valueset = false;
synchronized int get()
{ while(!valueset)
try {
   System.out.println("Consumer waiting");
   wait();
 } catch (InterruptedException e)
 { System.out.println("InterruptedException caught");
 }
 System.out.println("got:" +n);
 valueset =false;
 System.out.println("Intimate Producer");
 notify();
 return n;
 }
Synchronized void put(int n)
 { while(valueset)
 try {
     System.out.println("Producer waiting");
 catch (Interrupted exception e)
 System.out.println("InterruptedException caught");

 }
 this.n =n;
 valueset =true;
 System.out.println("Put" +n);
 System.out.println("Intimate Consumer");
 notify();
 }
}
```

```
clan Producer implement Runnable
Q q;
Producer(Q q)
{this.q=q;
new Thread(this,"Producer").start();
}
public void run()
{ int i=0;
while (i<15)
{q.put(i++);
}
}
}
~~class Consumer implements Runnable~~
class PCFixed
{public static void main(String args[])
{ Q q=new Q();
new Producer(q);
new Consumer(q);
System.out.println("Press control-c to stop");
}
}
```

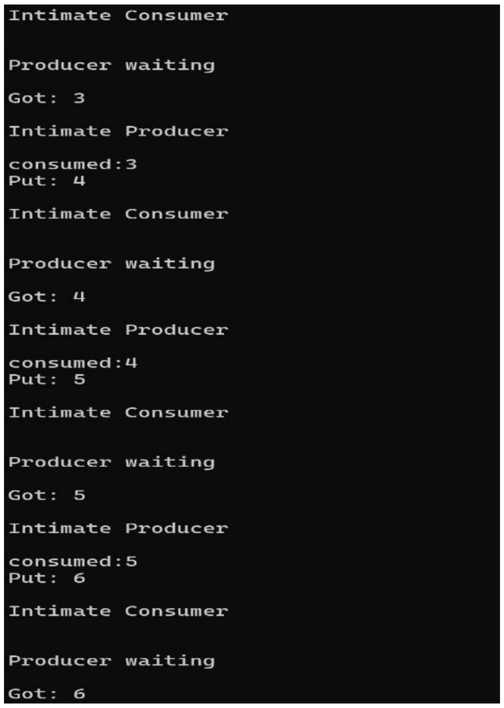
~~OUTPUT~~:

```
class Q { int n; boolean valueSet
= false;   synchronized
int get() {
while(!valueSet) try {
System.out.println("\nConsumer waiting\n"); wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught"); }
System.out.println("Got: " + n); valueSet = false;
System.out.println("\nIntimate Producer\n"); notify(); return
n;
}  synchronized void put(int n) { while(valueSet)
try {
System.out.println("\nProducer waiting\n"); wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
} this.n = n; valueSet
= true;
System.out.println("Put: " + n);
System.out.println("\nIntimate Consumer\n"); notify();
}  }
class Producer implements Runnable {
Q q;
Producer(Q q) { this.q = q; new Thread(this,
"Producer").start();
}  public void run() { int i =
0; while(i<15) {
q.put(i++);
}
}  }
class Consumer implements Runnable {
Q q;
Consumer(Q q) { this.q = q; new Thread(this,
"Consumer").start();  }
public void run() {  int
i=0; while(i<15) { int
r=q.get();
System.out.println("consumed:"+r); i++; }
}
}  class PCFixed {  public static void main(String
args[]) { Q q = new Q(); new Producer(q); new
Consumer(q);
System.out.println("Press Control-C to stop.");
}
}
```

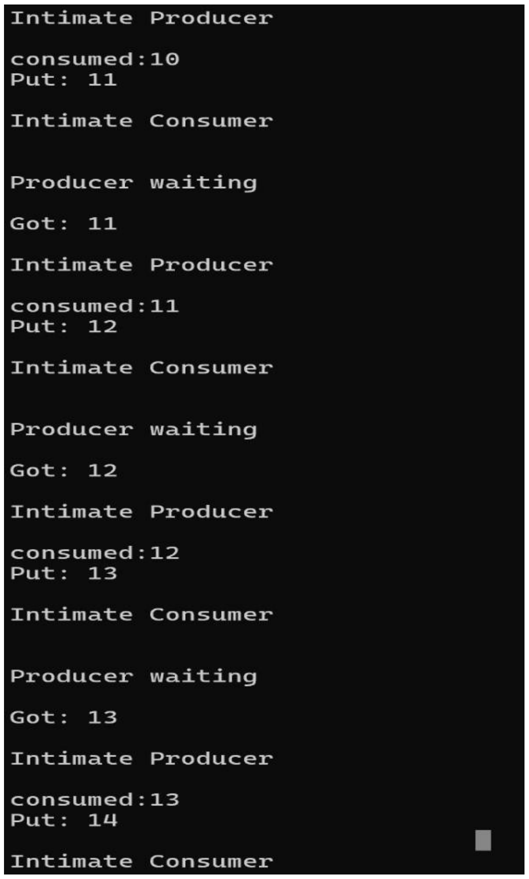
OUTPUT :



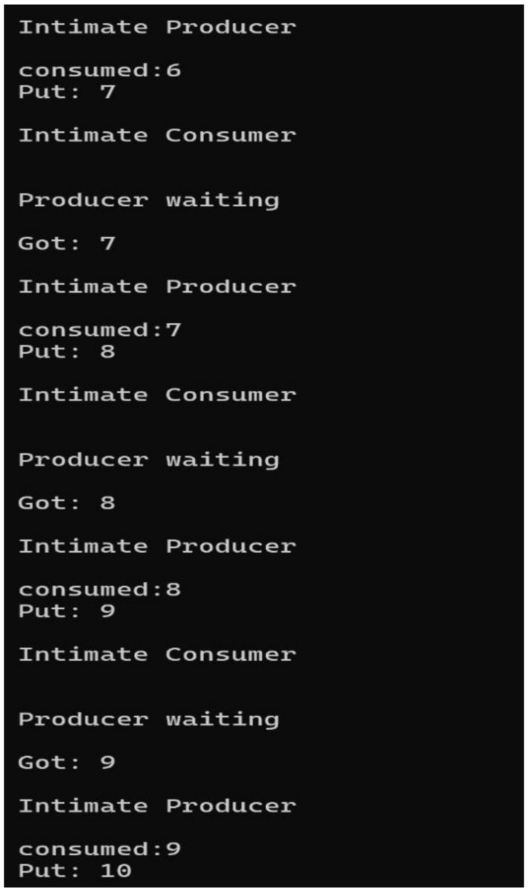
```
Press Control-C to stop.
Put: 0

Intimate Consumer


Producer waiting

Got: 0

Intimate Producer

Put: 1

Intimate Consumer


Producer waiting

consumed:0
Got: 1

Intimate Producer

consumed:1
Put: 2

Intimate Consumer


Producer waiting

Got: 2

Intimate Producer

consumed:2
Put: 3
```



```
Intimate Consumer


Producer waiting

Got: 3

Intimate Producer

consumed:3
Put: 4

Intimate Consumer


Producer waiting

Got: 4

Intimate Producer

consumed:4
Put: 5

Intimate Consumer


Producer waiting

Got: 5

Intimate Producer

consumed:5
Put: 6

Intimate Consumer


Producer waiting

Got: 6
```

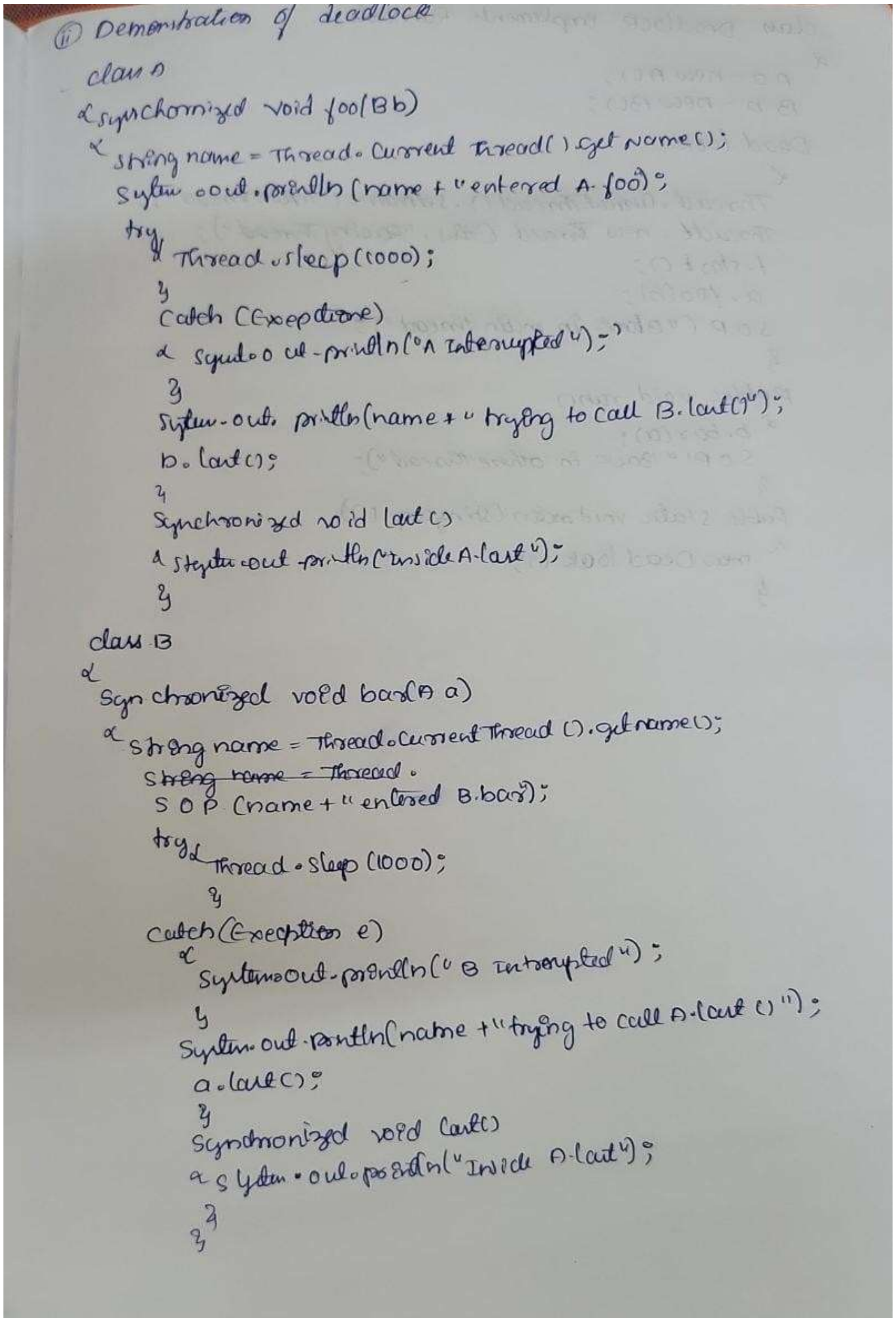


```
Intimate Producer

consumed:10
Put: 11

Intimate Consumer


Producer waiting

Got: 11

Intimate Producer

consumed:11
Put: 12

Intimate Consumer


Producer waiting

Got: 12

Intimate Producer

consumed:12
Put: 13

Intimate Consumer


Producer waiting

Got: 13

Intimate Producer

consumed:13
Put: 14

Intimate Consumer
```

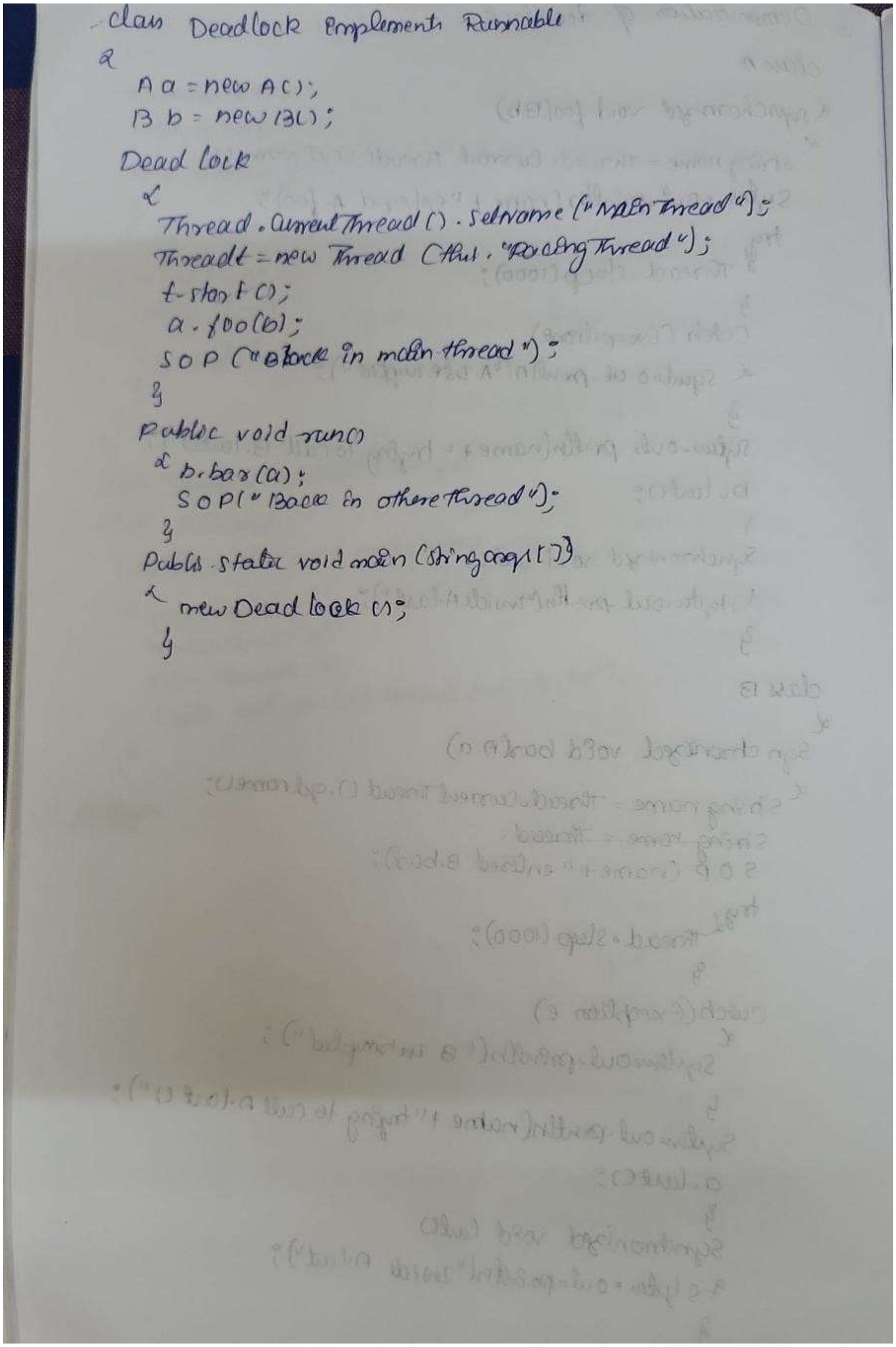


```
Intimate Producer

consumed:6
Put: 7

Intimate Consumer


Producer waiting

Got: 7

Intimate Producer

consumed:7
Put: 8

Intimate Consumer


Producer waiting

Got: 8

Intimate Producer

consumed:8
Put: 9

Intimate Consumer


Producer waiting

Got: 9

Intimate Producer

consumed:9
Put: 10
```

OBSERVATION:



```
(ii) Demonstration of deadlock
class A
{synchornized void foo(B b)
{
 string name = Thread. Current thread().get Name();
 system.out.println(name + "entered A.foo");
 try
 { Thread.sleep(1000);
  }
  catch (Exceptione)
  { system.out.println("A Interrupted");
   }
  system.out.println(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.Current Thread().getname();
  ~~string name = Thread.~~
  S O P (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");
    }
   }
```

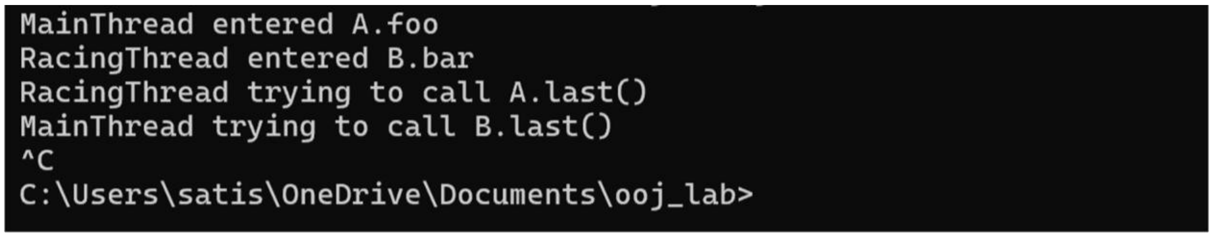


```
class Deadlock Implements Runnable
{
  A a = new A();
  B b = new B();
 Dead lock
  {
   Thread.CurrentThread().setname("MAin Thread");
   Threadt = new Thread (this, "RacingThread");
   t.start();
   a.foo(b);
   SOP ("Back in main thread");
  }
 public void run()
  { b.bar(a);
    SOP(" Back in othere thread");
  }
 Public static void main (String args[])
  { new Dead lock ();
  }
```

SOURCE CODE:

```
 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(); }        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(); a.foo(b); // get lock on a in this thread.      System.out.println("Back in main thread");
}   public void run() { b.bar(a); // get lock on b in other thread.
 System.out.println("Back in other thread");
  }
public static void main(String args[]) { new Deadlock(); }
              }
```

OUTPUT :



```
MainThread entered A.foo
RacingThread entered B.bar
RacingThread trying to call A.last()
MainThread trying to call B.last()
^C
C:\Users\satis\OneDrive\Documents\ooj_lab>
```