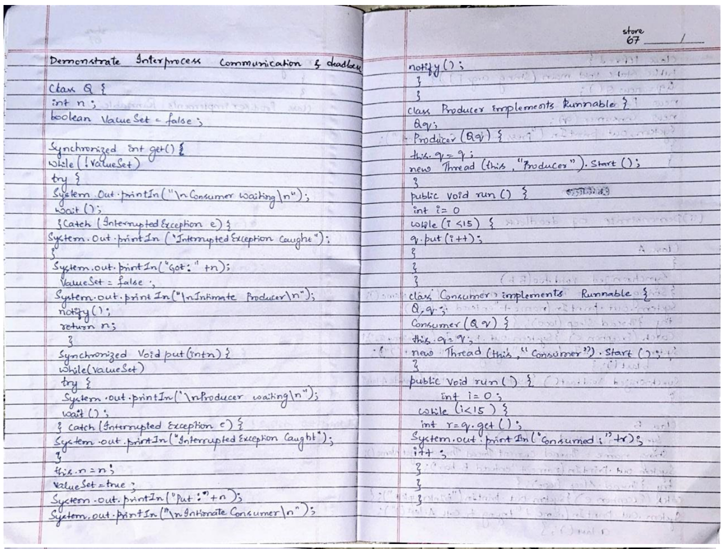
Program 10

Demonstrate Inter process Communication and deadlock

OBSERVATION:



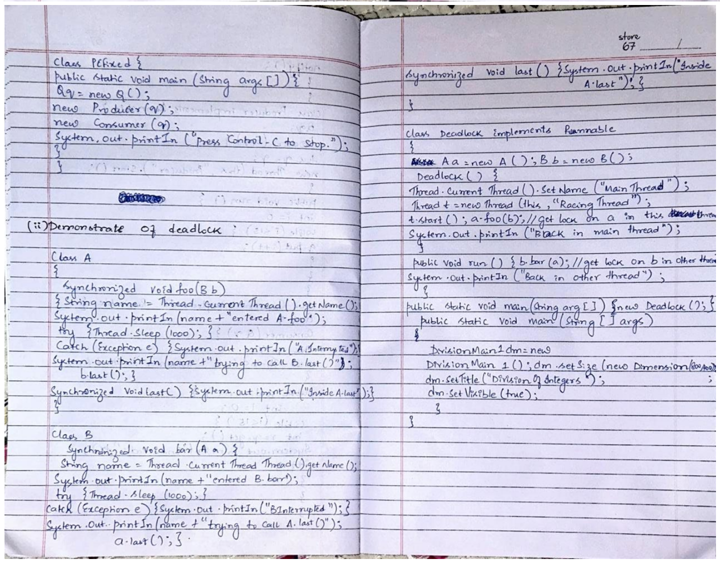
Demonstrate Interprocess Communication & deadlock

```
Class Q {
int n ;
boolean ValueSet = false ;

Synchronized int get() {
while (!ValueSet)
try {
System.Out.printIn("\nConsumer waiting\n");
wait ();
}Catch (InterruptedException e) {
System.Out.printIn ("Interrupted Exception Caught");
}
System.out.printIn("Got:" +n);
ValueSet = false ;
System.out.print In("\nIntimate Producer\n");
notify();
return n;
}
Synchronized Void put(intn) {
while(ValueSet)
try {
System.out.printIn("\nProducer waiting\n");
wait ();
} catch (Interrupted Exception e) {
System.out.printIn("Interrupted Exception Caught");
}
this.n = n;
ValueSet = true ;
System.out.printIn("Put :"+n);
System.out.printIn("\nIntimate Consumer\n");
```

store
67 ____/____

```
notify();
}
}
class Producer implements Runnable {
Qq;
Producer (Qq) {
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.print In("Consumed :"+r);
i++ ;
}
}
}
```



```
Class PCfixed {
public static void main (String args[]) {
Qq = new Q();
new Producer (q);
new Consumer (q);
System.out.printIn ("Press Control-C to stop.");
}
}
```

(ii)Demonstrate of deadlock

```
Class A
{
Synchronized void foo(B b)
{ String name = Thread.Current Thread ().getName();
System.out.printIn (name + "entered A.foo");
try {Thread.Sleep (1000); }
Catch (Exception e) {System.out.printIn ("A Intempted");}
System.out.printIn (name +"trying to call B.last()");
b.last(); }
Synchronized Void last() {System.out.printIn("Inside A.last");}
}

Class B
Synchronized Void bar (A a) {
String name = Thread.Current Thread Thread().get Name();
System.out.printIn (name +"entered B.bar");
try {Thread.sleep (1000); }
catch (Exception e) {System.Out.printIn("BInterrupted");}
System.Out.printIn (name + "trying to Call A.last()");
a.last(); }
```

store
67 ____/____

```
Synchronized void last () {System.Out.printIn("Inside
A.last");}
}

Class Deadlock implements Runnable
{
~~[illegible]~~ A a = new A(); B b = new B();
Deadlock () {
Thread.Current Thread ().Set Name ("Main Thread");
Thread t = new Thread (this, "Racing Thread");
t.start (); a.foo(b);//get lock on a in this ~~[illegible]~~ thread
System.Out.printIn ("Back in main thread");
}
public void run () { b.bar (a); //get lock on b in other thread
System.Out.printIn ("Back in other thread");
}
public static void main(String arg[]) {new Deadlock();}
public static void main (String [] args)
{
DivisionMain1 dm= new
DivisionMain 1 (); dm.setSize (new Dimension(800,400));
dm.SetTitle ("Division Of Integers");
dm.Set Visible (true);
}
}
```

CODE:

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.");

}
```

```
}
```

ii. Demonstration of 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();  {

      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);

  System.out.println("Back in other thread");

  }

public static void main(String args[]) { new Deadlock();


}
```