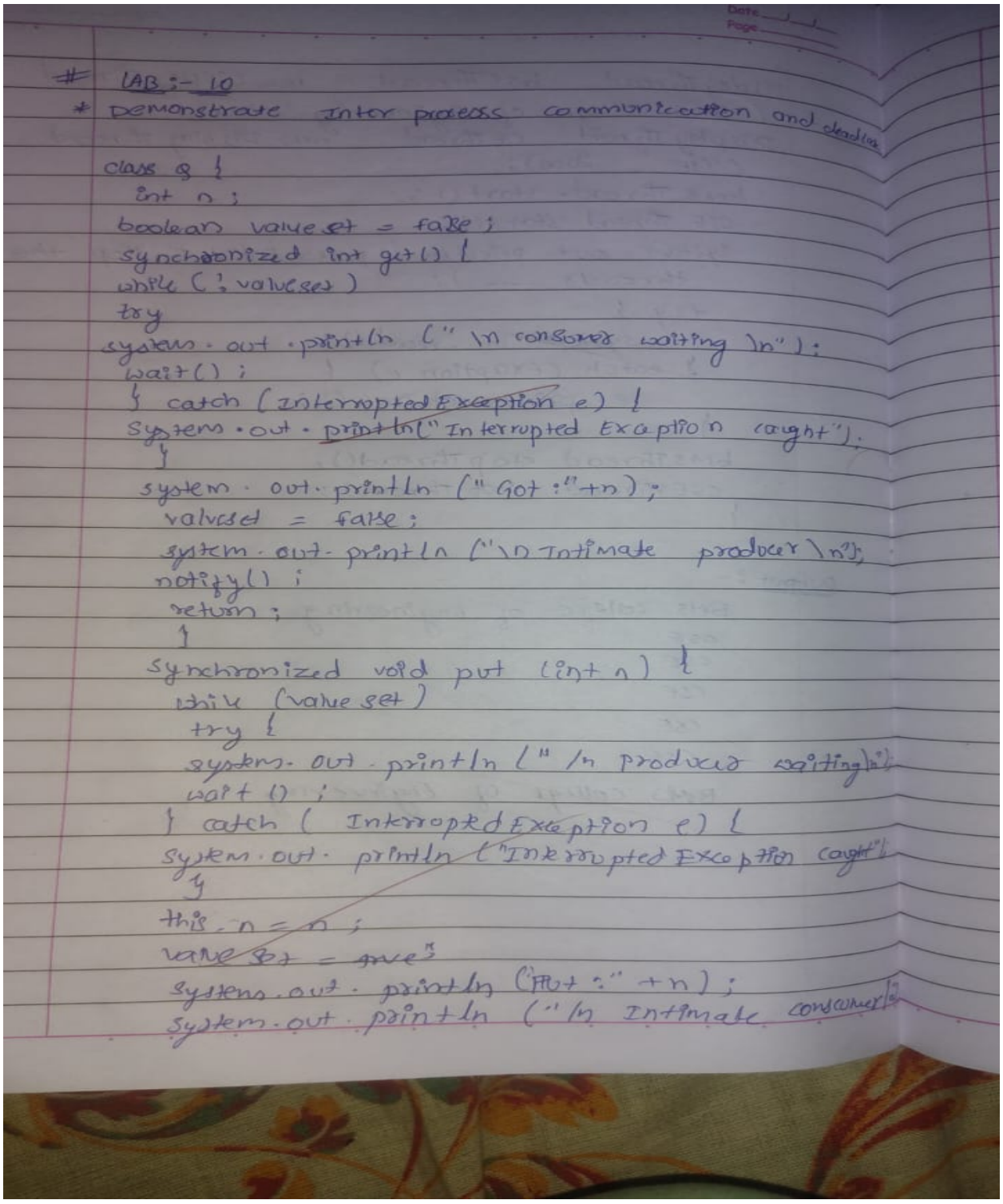
Lab 09 :

Inter process communication and deadlock :



LAB :- 10

* Demonstrate inter process communication and deadlock

```
class Q {
  int n;
  boolean valueSet = false;
  synchronized int get() {
  while (!valueSet)
  try
  System.out.println ("\n consumer waiting \n");
  wait();
  } catch (InterruptedException e) {
  System.out.println("Interrupted Exception caught");
  }
  system.out.println ("Got :" +n);
  valueSet = false;
  system.out.println ("\n Intimate producer \n");
  notify();
  return;
  }
  Synchronized void put (int n) {
  while (valueSet)
  try {
  System.out.println ("\n producer waiting");
  wait();
  } catch (InterruptedException e) {
  System.out.println ("Interrupted Exception caught");
  }
  this.n = n;
  valueSet = true;
  System.out.println ("Put :" +n);
  System.out.println ("\n Intimate consumer");
```

```
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()
{
Q q = new Q();
new producer(q);
new consumer(q);
System.out.println("Press 10 stop");
}
}
```

Output:

put:0
Get:0
put:1
Got:1
put:2
Got:2
put:3
Got:3
put:4
Got:4
Put:5
Got:5
put:6
Got:6
put:7
got:7
put:8
Got:8
put:9

put:10
Got:10
put:11
Got:11
put:12
Got:12
put:13
Got:13
put:14
Got:14

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.");

}

}
```

Output :

Put: 1

Got: 1

Put: 2

Got: 2

Put: 3

Got: 3

Put: 4

Got: 4

Put: 5

Got: 5