LAB-10 13/2/24

Q) Impumentation of produces and consumer

```
class Q {
int n ;
boolean valueset = false ;
synchronised int. get() {
while (!value set)
try {
System.out.priten("In Consumer waiting \n") ;

wait () ;
} catch (Interrupted Exception e) {
system. out. priten ("Interrupted Exception caught") ;
}
System .out. pritln ("Crt:" +n) ;
Value set = false
System. out. pritln ("In Intimate producer \n") ;

notify ();
return n ;
}
synchronized void put (int n) {
(value set)
try {
```

```
System.out.println("In Producer waiting \n");

wait();
} catch (InterruptedException e)
System.out.println("Interrupted Exception caught");
}

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;
```

```
}
public
& q
new Pr
new co
System.
}
}
o/p
Put: 1
Get: 1
Put: 2
Get: 2
Put: 3
Get: 3
Put: 4
```

```
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++;

}
}
}
public static void main(String args[]);
Q q=new Q();
new Producer(q);
new Consumer(q);
System.out.println("Press control.c to p");
}
}
```

O/P

```
Put: 1
Got: 1
Put: 2
Got: 2
Put: 3
Got: 3
Put: 4
Got: 4
```

Deadlocks

```
class A {

synchronized void (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();

}

class B {
synchronised 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");

}

a last();

}
void last() {
System.out.println("Inside A last");

}

}

class Deadlock implements Runnable

{
Aa = new A();
Bb = new B();

Deadlock() {
Thread t = new Thread(this, "Racing Thread");

}
public void run() {

to barla();

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

}
new Deadlock();

}

}
```

O/P

Main Thread entered A.foo

main Thread trying to call B.last()

Inside A'last

Back in main thread

Raiing Thread trying to call A.last()

Inside A.last

Back in other thread.