13/2/24

Lab Program 10

Implementation of producer and consumer

```
class Q {
int n;
boolean valueSet = false;
Synchronised 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;
}
synchronised void put(int n){
while(valueSet)
try{
System.out.println("\nProducer waiting\n");
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
\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);
```

```
        int 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.");
    }
}
```

O/P

```
Put: 1
Got: 1
Put: 2
Got: 2
Put: 3
Got: 3
Put: 4
Got: 4
Put: 5
Got: 5
```

Sakshi. B. R
1BM22CS233

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();
}
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();
}
void last(){
  System.out.println("Inside A.last");
}
}
class Deadlock implements Runnable
{
A a = new A();
B b = new B();
Deadlock(){
Thread.currentThread().setName("main Thread");
Thread t = new Thread(this, "Racing Thread");
t.start();
a.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();
}
}
```

o/p Main Thread entered A.foo
Racing thread entered B.bar
Main Thread trying to call B.last()
~~Hasta~~ Inside A.last
Back in main thread
Racing Thread trying to call A.last()
Inside A.last
Back in other thread.

Sakshi.B.R
1BM22CS233.

15-2-24