Lab - 10

```
class Q
{
 int n;
 synchronized ingel()
 {
  system.out.printlr("G ot: "+n);
  return n;
 }
synchronized void put(int n)
{
 this.n=n;
 System.out.println("Put: "+ n);
}
}
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+r);
   }
  }
 }
```

```
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();
    i++;
   }
  }
}
class PC
{
 public static void main (String args[])
 {
   Qq = new Q();
   new producer (q);
   new consumer (q);
   System. out. println (" Press control-c
                                  to stop.");
 }
}
```

<u>output:-</u>

```
put: 1          put: 2        put: 6
Got: 1          put: 3        put: 7
Got: 1          put: 4        put: 7
Got: 1          put: 5        Got: 7
```

correct implementation of Producer and consumer

```
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 (" put: " + n);
    notify();
    return n;
}
```

```
Synchronized void put(int n)
{  while (valueset)
   {
     try
     {
       system.out.println("\n producer
                          waiting \n");
       wait()
     }
     catch ( Intrreputed Exception caught");
   }
   this. n = n;
   valueset = true ;
   System. out. printn("Put; "+n);
   system. out. println ("Intimate consumer\n
   notify();
   }
}

class Producer implements runnable
{
   Q q;
   producer(Q q)
   {  this. q = q
      new Thred (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 Ctrl-C to stop");
   }
 }
```

DATE

output:

```
Put: 1
Got: 1
Put: 2
Got: 2
put: 3
Got: 3.
```

06/02/24

LAB - 10

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 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);
     ~~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();
}
}
```

output

```
Main Thread entered A. foo
Racing Thread entered B bar
main Thread trying to call B.last
Inside A.last
 Back in main Thread
 Inside A.last
 Back in other thread.
```

[illegible]
13/2/24