Lab Program - 10

13/02/24

```
class A {
    Synchronised 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 las() {
        System.out.println ("inside A.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");
    }
    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,"
                Racy 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 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
Racing thread trying to call A.last()
Inside A.last
Back in Other thread
```

Pcon.java

```
class Q {
    int n;
    boolean valueSet = false;
    synchronized int get() {
        while (!valueSet) {
            try {
                System.out.println("Consumer waiting.");
                wait();
            } catch (InterruptedException e) {
                System.out.println(e); }
```

```
System.out.println("got :" + n);
    value set = false;
System.out.println("Tell producer");
notify();
}
return 0;
}}
Synchronized void put(int n) {
    while (value set) {
    try {
System.out.println("Producer waiting")
wait();
} catch (Interrupted Exception e) {
    System.out.println(e);
}
this.n = n;
value set = true;
System.out.println("put:" + n);
System.out.println("Tell 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 < 3) {
```

```
        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<3) {
        int n = q.get();
      System.out.println("Consumed: "+n);
        i++;
      }
    }
  }
}
```

Output - 1

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

13.02.24