Lab - 10

```
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 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 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
```

06.02.24

ProCon.java

```
class Q {
    int n;
    boolean value set = false;
    synchronised int get() {
        while (!value set) {
            try {
                System.out.println("Consumed Waiting ");
                wait();
            } catch (InterruptedException e) {
                System.out.println(e); }
```

```
        System.out.println("Got: " + n);
        value set = false;
        System.out.println("Tell Producer");
        notify();
    }
    return n;
}

synchronized void put(int n) {
    while (value set) {
        try {
            System.out.println("Producer Waiting");
            wait();
        } catch (InterruptedException 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

13/2/24