13/02/24

LAB - 10 a

Demonstrate Inter Process Communication

```
class Q{
    int n;
    boolean valueSet = false;
    synchronised int get(){
        while (!valueSet)
            try{
                System.out.println("\nConsumer Waiting");
                wait();
            } catch (InterruptedException e){
                System.out.println ("\n Interupted
                        Exception Caught");
            }
        System.out.println("Got : " +n);
        valueSet = false;
        System.out.println ("\n Intimate Producer");
        notify();
        return n;
    }
    synchronised void put(int n){
        while (valueSet)
            try{
                System.out.println("\nProducer Waiting");
                wait();
            } catch (InterruptedException e){
                System.out.println("\n Interupted
                        Exception caught");
            }
        this.n = n;
        valueSet = true;
        System.out.println("Put: "+n);
        System.out.println("Intimate 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 < 4){
            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<4){
            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 control C to stop");
    }
}
```

OUTPUT:

```
Put : 0
Got : 0
Put : 1
Got : 1
Put : 2
Got : 2
Put : 3
Got : 3
Put : 4
Got : 4
```

13/2/24

SURYA Gold

Date______ Page______

LAB - 10 b

Demonstrate 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 Interupted");
     }
     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("MainThead");
            Thread t= new Thread(this, "RacingThread");
            t.start();
            a.foo(b);
            System.out.println("Back in mainthread");
        }
        public void run(){
            b.bar(a);
            System.out.println("Back in other thread");
        }
        public static void main(String args[]){
            new Deadlock();
        }
    }
```

OUTPUT:

```
Mainthread entered A.foo
Racingthread entered B.bar
Racing thread trying to call A.last()
Inside A.last
Back in otherhead
MainThread trying to call B.last()
Inside A.last
Back in main thread
```

13/2/2024