6/2/24

Lab Prgm - 10.

19. Demonstrate inter process communication and deadlock.

```
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("InterruptedException caught");
  }
System.out.println("Got: " + n);
valueSet = false;
System.out.println("\n Intimate Producer\n");
notify();
return n;
}
Synchronized void put(int n){
 while(valueSet)
 try{
  System.out.println("\n Producer waiting \n");
  wait();
 } catch(InterruptedException e);caught");
   { System.out.println("InterruptedException caught");
   }
 this.n = n;
 valueSet = true;
```

```
        System.out.println("Put: "+n);
        System.out.println("In 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);
     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

Deadlock.

```
class A {
  synchronized void for(B b){
    String name = Thread.currentThread().getName();
    SOP (name + " enteredA.for");
    try {
      Thread.sleep(1000);
      }
    catch(exception e)
      {
        SOP ("AInterrupted");
        SOP(name + " trying To call B.last()");
        b.last();
      }
    void last (){
      SOP ("Inside A.last");
    }
}
class B {
  Synchronized vair bar (A a) {
    String name = Thread.CurrentThread().getName();
    SOP (name + " Entered B.bar");
    try {
      Thread.sleep(1000);
    }
    catch (exception e)
      {
        SOP("B.Interrupted");
      }
  SOP (name + " trying To call A.last()");
        a.last(); }
```

```
void last(){
    SOP("Inside A.last");
  }
}
class Deadlock implements Runnable
{
    A a = new A();
    B b = new B();
    Deadlock(){
     Thread.currentThread().setName("MainThread");
     Thread t = new Thread(this, "Racing thread");
     t.start();
     a.foo(b);
     SOP("Back in mainthread.");
    }
  public void run(){
     b.bar(a);
     SOP("Back in Other Thread");
  }
  public static void main(String arg()){
     new Deadlock();
  }
}
```

O/P: MainThread entered A.foo
Racing thread entered B.bar
MainThread trying to call B.last()
Inside A.last
Back in MainThread
Racing thread trying to call A.last()
Inside A.last
Back in Other Thread.

13.02.24