13/2/24

Date :
Page No :

LAB -10 (b)

Demonstrate Deadlock

```
class A
{
    synchronized void fruits (B b)
    {
        string name = Thread.currentThread().
                                    getName();
        System.out.println (name + "entered A.fru
        try {
        Thread.sleep(1000);
        } catch (Exception e){
          System.out.println(name + "trying to
                call B.last()");
        b.last();
    }
    void last(){
        System.out.println( "Inside A.last");
    }
}

class B
{
    synchronized void veggies(A a)
    {
        String name = Thread.currentThread().
                                    getName();
        System.out.println (name + "entered
                                B.veggies ");
        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,
                          "RunningThread");
      t.start();
      a.fruits(b);
      System.out.println("Back in main
                        thread");
    }
    public void run(){
      b.veggies(a);
      System.out.println("Back in other
                        thread");
    }
```

```
public static void main(String arg[])
{
    new Deadlock();
}
}
```

OUTPUT -

Main Thread entered A.fruits
Running Thread entered B.veggies
Mainthread trying to call B.last()
Inside A.last
Back in main thread
RunningThread trying to call A.last()
Inside A.last
Back in other thread

13/2/24