Lab 8. programs:-

```
import java.util.*;
import java.io.*;

class B extends Thread {
            public void run () {
              try {
                    for (int i=0; i<3; i++){
                    System.out.println ("BMS");
                    Thread.sleep (1000);
                  }
                   catch (Interrupt Exception e)
                  {
                       System.out.println (e);
                  }
              }
  }
class C extends Thread {
     public void run() {
      try {
            for (int i=0;i<3;i++)
            {
             System.out.println ("CSE");
              Thread.sleep (2000);
            }
      }.
   catch (InterruptionExecution e) {
            System.out.println (e);
         }
      }
  }.
```

```
class Thread Main {
    public stati void main (String args [])
  {
        Bb = new B();
        Cc = new C();
        b. start ();
        c. start ();
    }
}
```

output :-

```
BMS
CSE
CSE
CSE
Bms
Bms
```

x — x — x — x

```
class A {
   Synchronized void foo (B b)
   {
       String name = Thread . current Thread () . get Name ();
       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 {
        synchronizied void bus (A a)
        {
            String name = Thread. current Theead () get Name ();
            System.out.println (name + "enter 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 implement s
   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 void run()
{
   b.bar(a);
   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.
```

13/2/24