Page No.
Date 13/02/24

Lab 9 programs:-

Thread main:-

```
import java.io.*;

class B extends Thread{
    public void run(){
        try{
            for(int i=0; i<3; i++){
                System.out.println("Bms");
                Thread.sleep(10000);
            }
        }catch(InterruptedException 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(20000);
            }
        }catch(InterruptedException e){
            System.out.println(e);
        }
    }
}
class Threadmain{
    public static void main(String args[]).
```

```
{
    B b = new B();
    C c = new C();
    b.start();
    c.start();
  }
}
```

Deadlock.java :-

```
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 {
    S.O.P("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);
    S.O.Pln("Back in thread");
  }
    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.

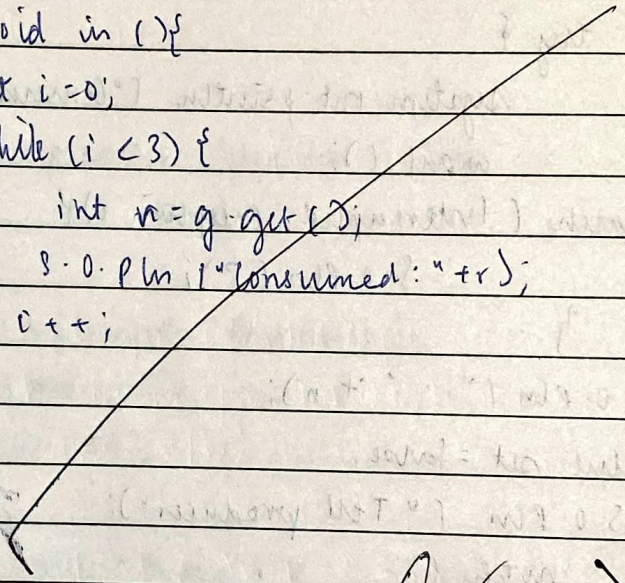
Page No.
Date

Pro. con Java.

```
class Q {
    int n;
    boolean valueSet = false;
    synchronized int get(){
        while(!valueSet){
            try {
                System.out.println("Consumed waiting");
                wait();
            } catch (InterruptedException e){
                S.o.Pln(e);
            }
        S.o.Pln(" : " + n);
        valueSet = false;
        S.o.Pln("Tell producer");
        notify();
    }
    return n;
}
}
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++);
        }
    }
}
```

Page No.
Date

```
Class consumer implements Runnable {
    Q q;
    Consumer (Q q) {
        this.q = q;
        New Thread (this, "Consumer").start();
    }
    public void in () {
        int i = 0;
        while (i < 3) {
            int n = q.get();
            S.O.Pln ("Consumed: " + r);
            i++;
        }
    }
}
```



Output:

Put : 1
get : 1
Put : 2
get : 2
Put : 3
get : 3