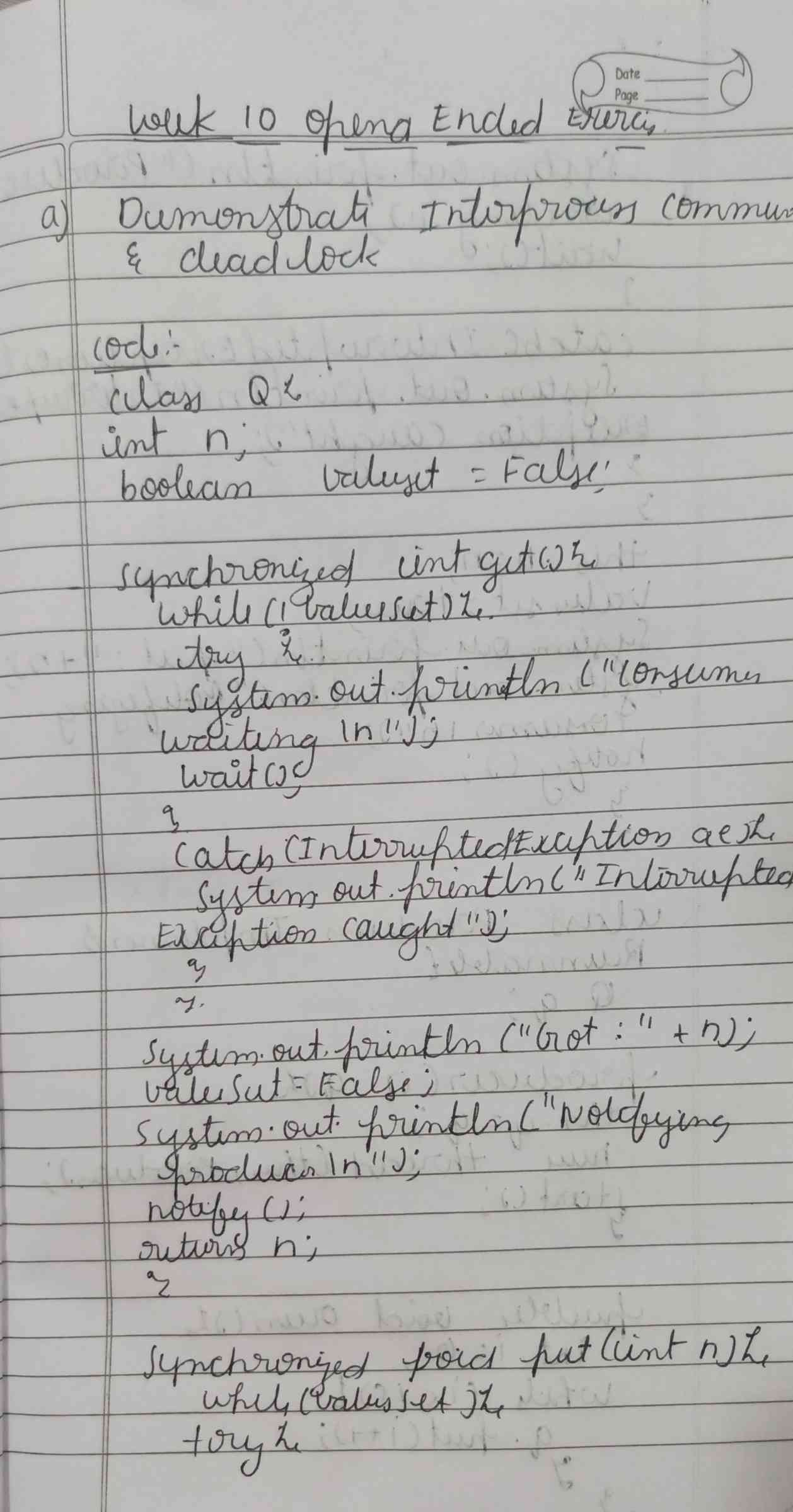
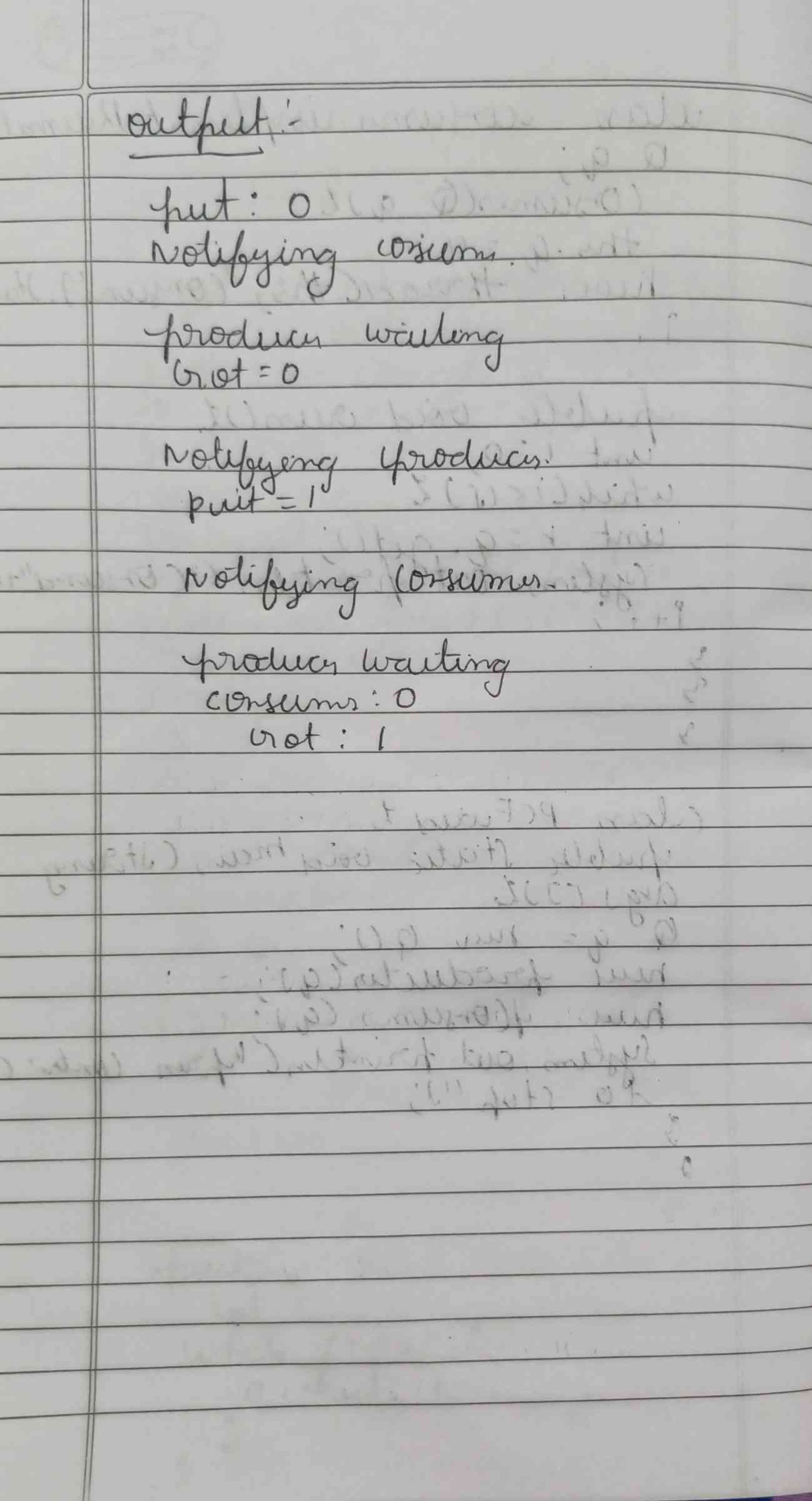
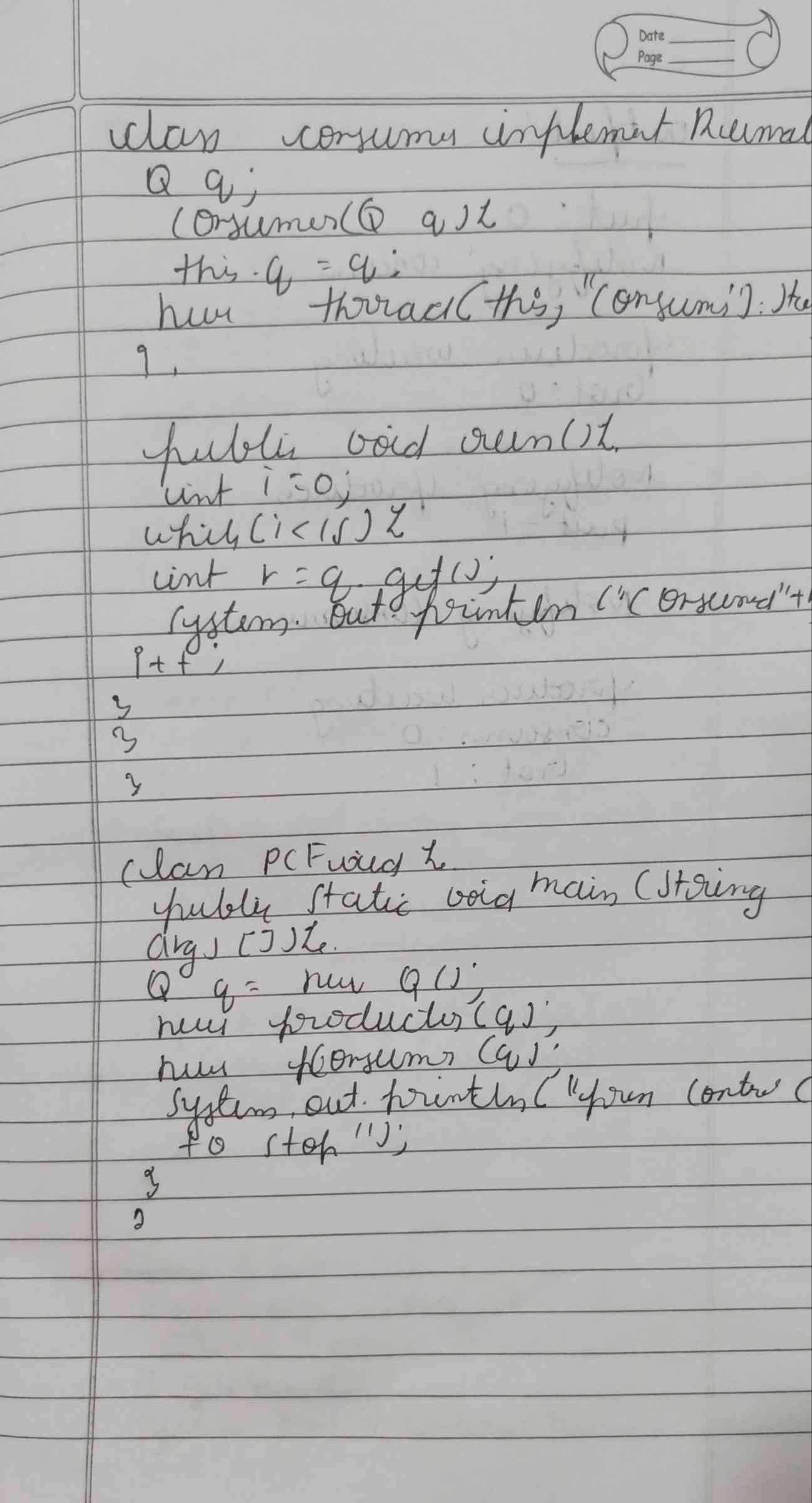
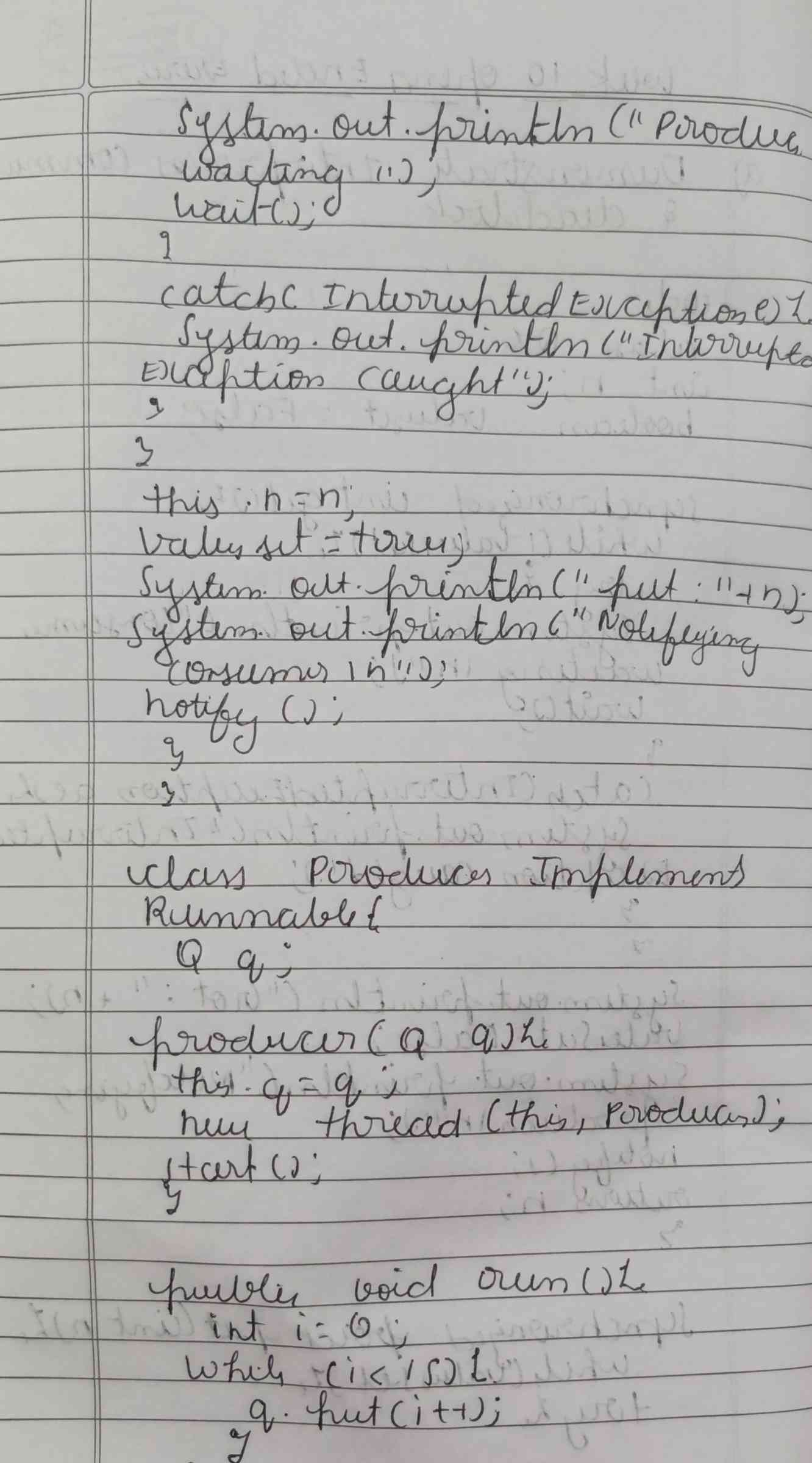
**LABORATORY PROGRAM – 10**

Demonstrate Inter process Communication and deadlock.

**OBSERVATION :**

****

****

**CODE :**

**class Q {**

**int n;**

**boolean valueSet = false;**

**synchronized int get() {**

**while (!valueSet) {**

**try {**

**System.out.println("\nConsumer waiting\n");**

**wait();**

**} catch (InterruptedException e) {**

**System.out.println("InterruptedException caught");**

**}}**

**System.out.println("Got: " + n);**

**valueSet = false;**

**System.out.println("\nNotifying Producer\n");**

**notify();**

**return n;}**

**synchronized void put(int n) {**

**while (valueSet) {**

**try {**

**System.out.println("\nProducer waiting\n");**

**wait();**

**} catch (InterruptedException e) {**

**System.out.println("InterruptedException caught"); } }**

**this.n = n;**

**valueSet = true;**

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

**System.out.println("\nNotifying 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++); }**

**System.out.println("Producer finished."); } }**

**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++; }**

**System.out.println("Consumer finished.");**

**} }**

**class PCFixed {**

**public static void main(String args[]) {**

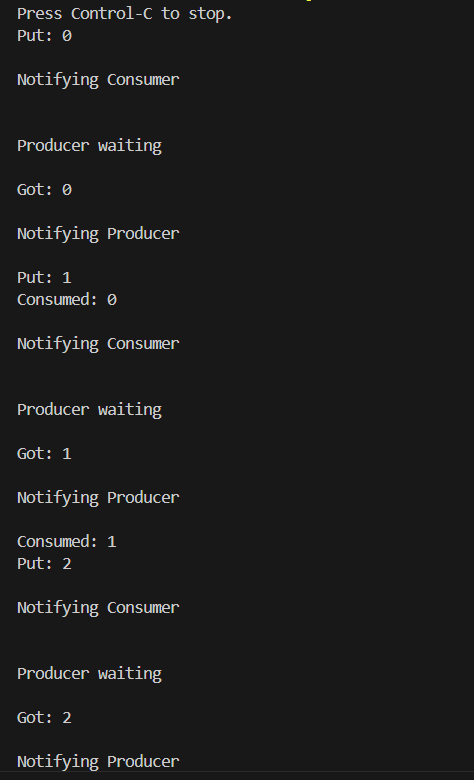
**Q q = new Q();**

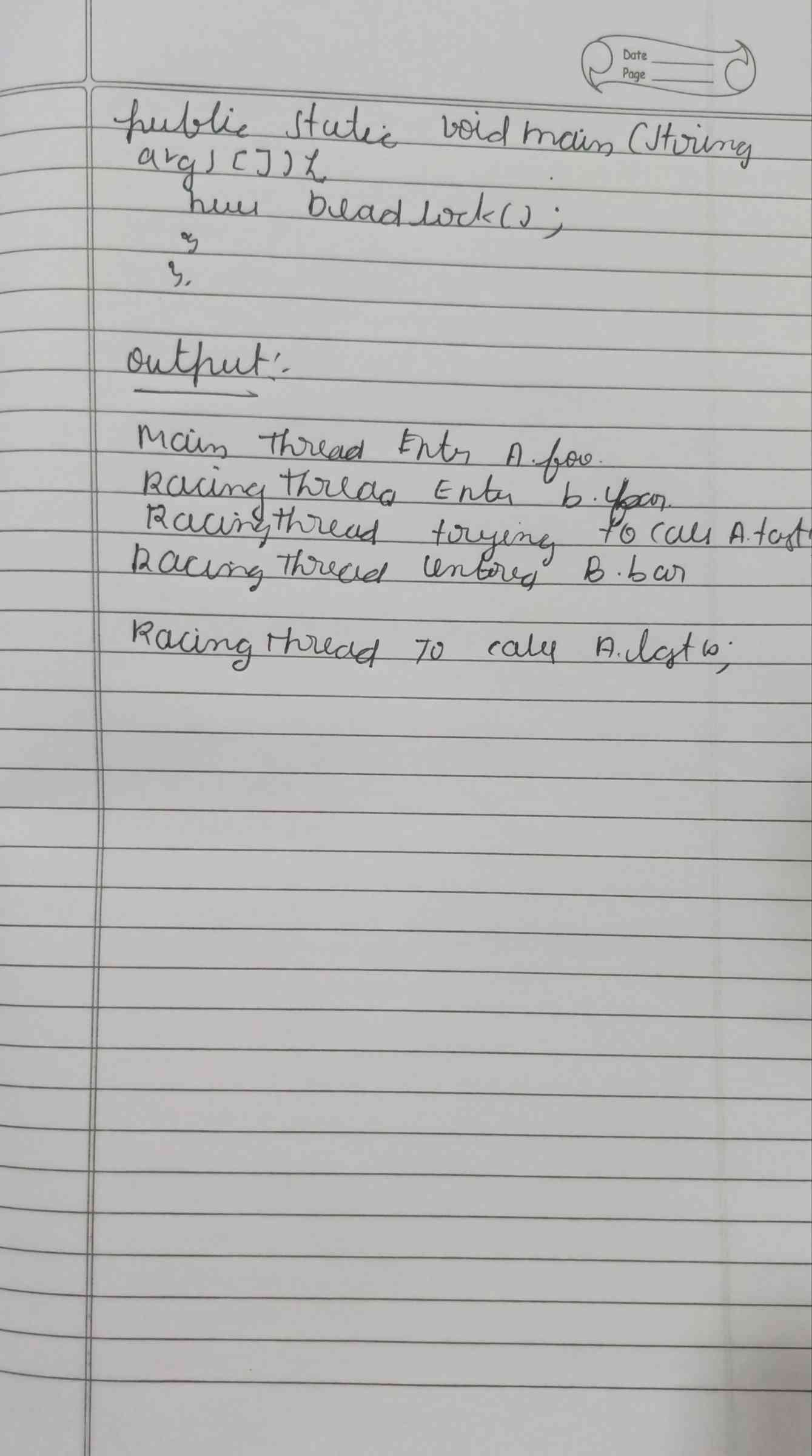
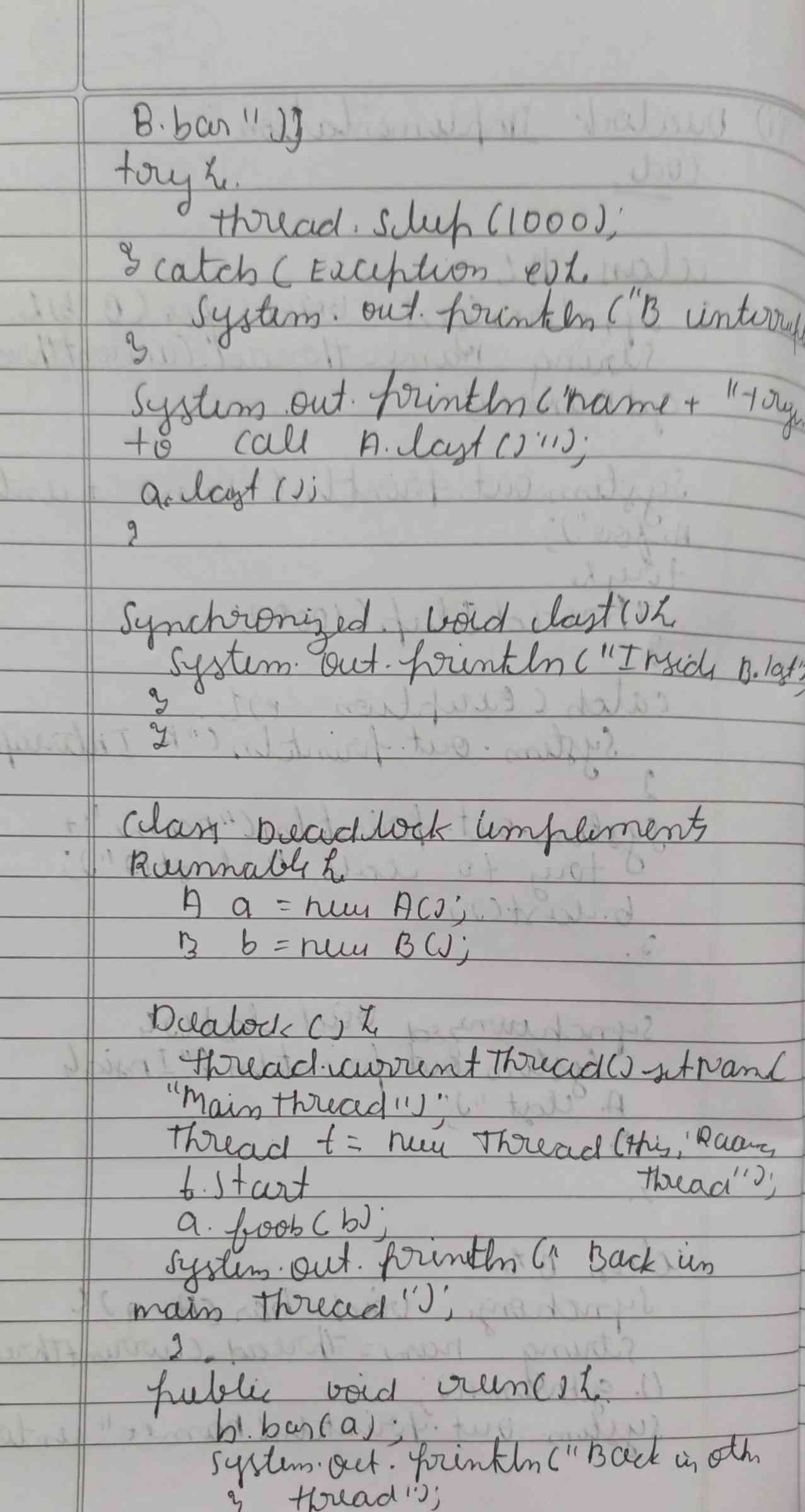
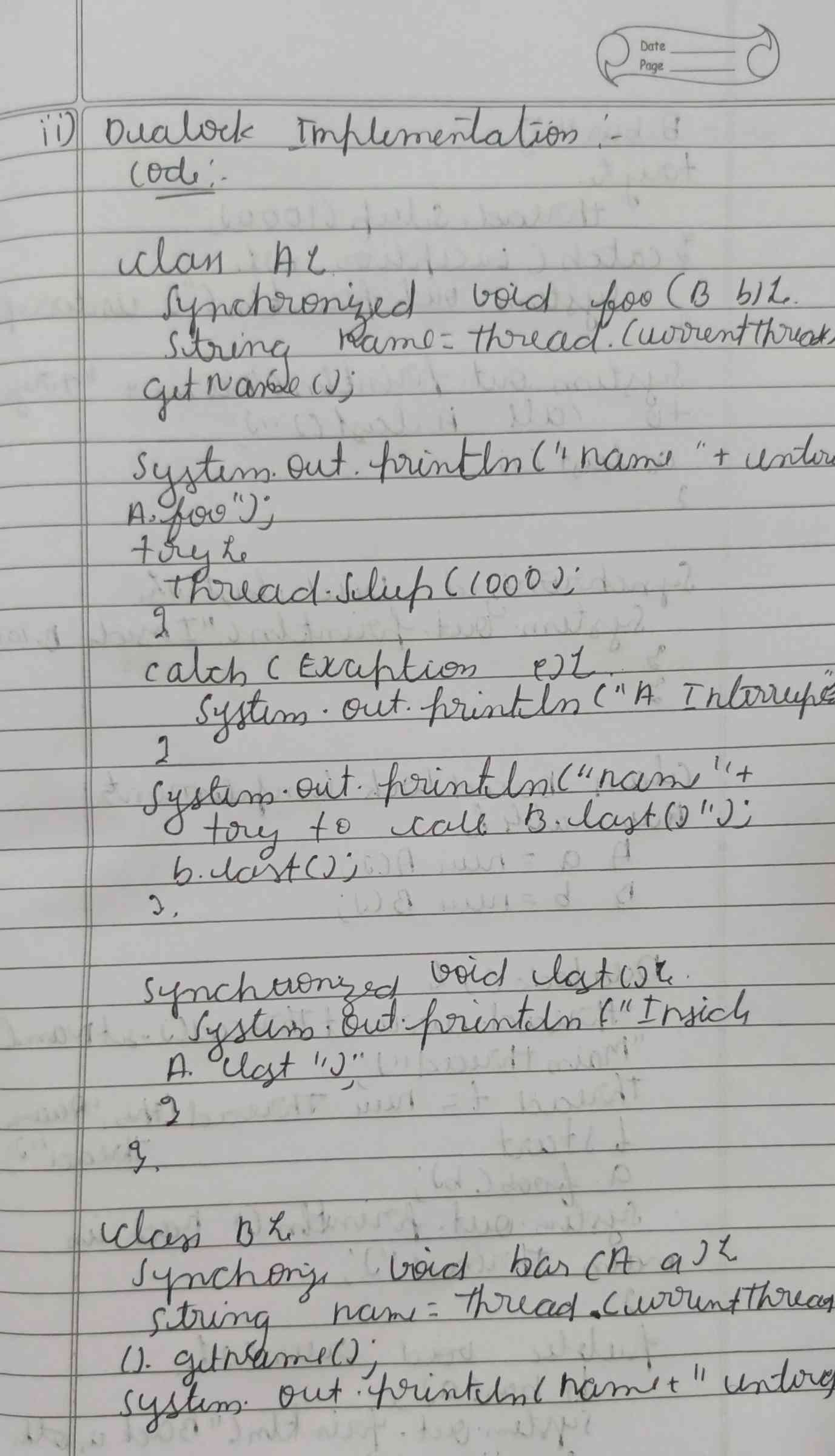
**new Producer(q);**

**new Consumer(q);**

**System.out.println("Press Control-C to stop."); } }**

**OUTPUT :**

****

****

**CODE :**

**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();**

**}**

**synchronized 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 Interrupted");**

**}**

**System.out.println(name + " trying to call A.last()");**

**a.last();**

**}**

**synchronized void last() {**

**System.out.println("Inside B.last");**

**}**

**}**

**class Deadlock implements Runnable {**

**A a = new A();**

**B b = new B();**

**Deadlock() {**

**Thread.currentThread().setName("MainThread");**

**Thread t = new Thread(this, "RacingThread");**

**t.start();**

**a.foo(b);**

**System.out.println("Back in main thread");**

**}**

**public void run() {**

**b.bar(a);**

**System.out.println("Back in other thread");**

**}**

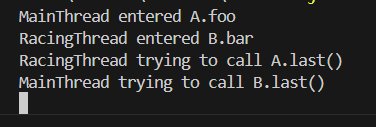
**public static void main(String args[]) {**

**new Deadlock();**

**}**

**}**

**OUTPUT :**

****