

# **B.M.S COLLEGE OF ENGINEERING BENGALURU**

Autonomous Institute, Affiliated to VTU



## **LAB REPORT 23CS3PCOOJ**

Submitted in partial fulfilment of the requirements for Lab Bachelor of Engineering

in

Computer Science and Engineering

Submitted by:

**PRANAV ANANTHA RAO**

**(1BM22CS201)**

Department of Computer Science and Engineering,

B.M.S College of Engineering,

Bull Temple Road, Basavanagudi, Bangalore, 560 019 2023-2024.

## **INDEX**

<b>Sl.No.</b>	<b>Title</b>	<b>Date</b>
1	Complete scanned Observation Book	12/12/2023 - 20/02/2024
2	Lab 1	12/12/2023
3	Lab 2	19/12/2023
4	Lab 3	26/12/2023
5	Lab 4	02/01/2024
6	Lab 5	09/01/2024
7	Lab 6	16/01/2024
8	Lab 7	23/01/2024
9	Lab 8	30/01/2024
10	Lab 9	06/02/2024
11	Lab 10	20/02/2024

→ Develop a java program that prints all real solns to the quadratic equation  $ax^2 + bx + c = 0$ . If  $b^2 - 4ac < 0$ , display a message that there are no real solns.

```

import java.util.Scanner;
class quadratic {
    public static void main (String args[]) {
        Scanner sc = new Scanner (System.in);
        System.out.println ("Pranav Anantha Rao");
        System.out.println ("IBM22CS201");
        System.out.println ("Enter coefficients : ");
        int a, b, c;
        double r1, r2;
        a = sc.nextInt();
        b = sc.nextInt();
        c = sc.nextInt();
        while (a == 0) {
            System.out.println ("Not a quadratic eqn");
            System.out.println ("Enter non-zero value of a");
            a = sc.nextInt();
        }
        int d = b * b - 4 * a * c;
        if (d == 0) {
            r1 = (-b) / (2 * a);
            System.out.println ("Real and equal roots ");
            System.out.println ("Roots are " + r1);
        } else if (d > 0) {
            r1 = ((-b) + Math.sqrt(d)) / (double) (2 * a);
            r2 = ((-b) - Math.sqrt(d)) / (double) (2 * a);
        }
    }
}

```

System.out.println("Roots are real and distinct");  
 System.out.println("Roots are "+r1+", "+r2);

} else if (d < 0) {

$$r1 = (-b) / (2 * a);$$

$$r2 = Math.sqrt(-d) / (2 * a);$$

System.out.println("Roots are Imaginary");

System.out.println("Root 1 = "+r1+" + i "+r2);

System.out.println("Root 2 = "+r1+" - i "+r2);

}

}

Output :

Pranav Anantha Rao

1BM20CS201.

Enter coefficients:

1

2

1

Roots are real and equal

$- d^2 = b^2$

Roots are -1.

Enter coefficients:

1

1

2

Roots are Imaginary

Root 1 = 0.0 + i 1.322875

$$\text{Root } \alpha = 0.0 - i 1.322875$$

Enter coefficient of a, b, c

1

-3

2

Roots are real and distinct

Roots are 2.0 1.0

Enter coefficients

0

4

2

Not a quadratic equation

Enter a non zero value of a:

2.

Roots are real and equal

Roots are -1.0

✓  
red ✓

19/12

Q.

Develop a java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

```
import java.util.Scanner;
```

```
class Subject {
```

~~int subjectMarks;~~~~double SGPA;~~~~int credits;~~~~int grade;~~

```
}
```

```
class Student {
```

```
String name, usn;
```

```
double SGPA;
```

```
Scanner s;
```

```
Subject [] subject;
```

```
Student () {
```

```
int i;
```

```
subject = new Subject [9];
```

```
for (int i = 0; i < 9; i++) {
```

~~subject[i] = new Subject();~~

```
y
```

~~s = new Scanner (System.in);~~

```
}
```

```
void getStudentDetails () {
```

```
System.out.println ("Enter name: ");
```

```
name = s.nextLine ();
```

```
System.out.println ("Enter USN : ");
USN = s.nextLine();
```

{

```
void getMarks () {
```

```
for (int i = 0; i < 9; i++) {
```

```
System.out.println ("Enter marks for " +  
(i + 1) + "th Subject : ");
```

```
subject[i].subjectMarks = s.nextInt();
```

```
System.out.println ("Enter credits for " +  
(i + 1) + "th subject");
```

```
subject[i].credits = s.nextInt();
```

```
if (subject[i].subjectMarks > 100 ||
```

```
subject[i].subjectMarks < 0) {
```

~~while (subject[i].subjectMarks < 0 ||  
subject[i].subjectMarks > 100) {~~

```
System.out.println ("Invalid marks  
enter again : ");
```

```
subject[i].subjectMarks = s.nextInt();
```

}

```
if (subject[i].subjectMarks >= 90) {
```

```
subject[i].grade = 10;
```

```
else if (subject[i].subjectMarks >= 80) {
```

```
subject[i].grade = 9;
```

~~else if (subject[i].subjectMarks >= 70) {~~

```
subject[i].grade = 8;
```

~~else if (subject[i].subjectMarks >= 60) {~~

```
subject[i].grade = 7;
```

~~else if (subject[i].subjectMarks >= 50) {~~

```
subject[i].grade = 6;
```

~~else if (subject[i].subjectMarks >= 40) {~~

```
    subject[i].grade = 5;  
} else if (subject[i].subjectMarks >= 0) {  
    subject[i].grade = 0;  
}
```

{

```
void SGPA() {
```

```
    int creditsxgrade = 0;
```

```
    int total credits = 0;
```

```
    for (int i = 0; i < 9; i++) {
```

 ~~creditsxgrade += (subject[i].grade \* subject[i].credit);~~ ~~total credits += subject[i].credit;~~

}

 ~~SGPA = creditsxgrade / total credits;~~

```
void display() {
```

```
    System.out.println("Name: " + name);
```

```
    System.out.println("USN: " + usn);
```

```
    System.out.println("SGPA: " + SGPA);
```

}

}

```
class Main {
```

~~public static void main (String args[]) {~~ ~~Student s1 = new Student();~~ ~~s1.getStudentDetails();~~ ~~s1.getMarks();~~ ~~s1.SGPA();~~ ~~s1.display();~~

}

}

Output:

Enter name: Pranav Anantha Rao

Enter USN: 1BM32CS201

Enter marks and credits:

95

4

97

4

97

3

96

3

90

3

88

1

98

1

92

1

Name: Pranav Anantha Rao

USN: 1BM32CS201

SGPA: 9.95

Fri. 19/12/23

26/12/23

3. Create a book class which has 4 members : name, author, price, num pages. Include constructor to set value for members. Include methods to set and get details of objects. Include toString() method to display details.

```
import java.util.Scanner;  
class Book {  
    String name, author;  
    int price, num_pages;  
  
    Book (String name, String author, int price, int num_page)  
    {  
        this.name = name;  
        this.author = author;  
        this.price = price;  
        this.num_pages = num_pages;  
    }  
}
```

```
public String toString ()  
{  
    String n, a, p, N;  
    n = "Name of Book : " + name + "\n";  
    a = "Author of Book : " + author + "\n";  
    p = "Price of Book : " + price + "\n";  
    N = "Number of pages : " + num_pages + "\n";  
    return n + a + p + N;  
}
```

```
class Main {
```

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

```
        Scanner sc = new Scanner (System.in);
```

```
        System.out.println ("Enter no of books: ");
```

```
        int n = sc.nextInt();
```

```
        Book b[] = new Book [n];
```

```
        String name, author;
```

```
        int price, num;
```

```
        sc.nextLine();
```

~~```
        for (int i = 0, i < n; i++) {
```~~~~```
            System.out.println ("In Enter name of Book: ");
```~~~~```
            name = sc.nextLine();
```~~~~```
            System.out.println ("In Enter author's name: ");
```~~~~```
            author = sc.nextLine();
```~~~~```
            System.out.println ("In Enter price: ");
```~~~~```
            price = sc.nextInt();
```~~~~```
            System.out.println ("In Enter num of pages: ");
```~~~~```
            num = sc.nextInt();
```~~~~```
            b[i] = new Book (name, author, price, num);
```~~

```
}
```

~~```
        System.out.println ("Book Details: ");
```~~~~```
        for (int i = 0; i < n; i++) {
```~~~~```
            System.out.println (b[i].toString());
```~~

```
}
```

```
}
```

Output:

Enter number of books:

1

Enter name of book:

Bible

Enter author's name:

God

Enter price:

1000

Enter number of pages:

1000

Book details:

Name of the book: Bible

Author of book: God

Price of Book: 1000

Number of pages: 1000

- Pranav Anantha Rao

1BM22CS201

02/01/24

4. Develop a java program to create an abstract class named shape that contains two integers & an empty method named printArea(). Provide 3 classes named Rectangle, Triangle, Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea() that prints the area of the given shape.

```
import java.util.Scanner;
```

```
class InputScanner {
```

```
    Scanner sc = new Scanner(System.in);
```

```
}
```

```
abstract class Shape {
```

```
    int l1, l2;
```

```
    Shape(int a, int b) {
```

```
        l1 = a;
```

```
        l2 = b;
```

```
}
```

```
    abstract double printArea();
```

```
}
```

```
class Rectangle extends Shape {
```

```
    Rectangle(int a, int b) {
```

```
        super(a, b);
```

```
}
```

```
    double printArea() {
```

```
        System.out.println("Area of rectangle is ");
```

```
        return l1 * l2;
```

```
}
```

{

```
class Triangle extends Shape {
```

```
    Triangle (int a, int b) {
```

```
        super (a, b);
```

}

```
    double printArea () {
```

```
        System.out.println ("Area of Triangle is ");
```

```
        return 0.5 * l1 * l2;
```

}

{

```
class Circle extends Shape {
```

```
    Circle (int a, int b) {
```

```
        super (a, b);
```

}

```
    double printArea () {
```

```
        System.out.println ("Area of circle is ");
```

```
        return 3.14 * l1 * l1 / 7;
```

}

{

```
class Main extends InputScanner {
```

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

```
        int a, b;
```

```
        System.out.println ("Enter sides of rectangle ");
```

```
        a = sc.nextInt ();
```

```
        b = sc.nextInt ();
```

```
        Rectangle r = new Rectangle (a, b);
```

```
        System.out.println ("Enter base and height of triangle ");
```

```
        a = sc.nextInt ();
```

```
b = sc.nextInt();
Triangle t = new Triangle(a, b);
System.out.println("Enter radius of circle");
a = sc.nextInt();
Circle c = new Circle(a, 1);
Shape figref;
figref.a = r;
System.out.println(figref.printArea());
figref = t;
System.out.println(figref.printArea());
figref = c;
System.out.println(figref.printArea());
```

{}

Output:

Enter sides of rectangle:

10 10

Enter base &amp; height of triangle:

10 10

Enter radius of circle

25

Area of rectangle is  
100.0Area of triangle is  
50.0Area of circle is  
1964.1857

Enter sides of rectangle

10 10

Enter sides of triangle

10 -9

~~Invalid input~~

Ans  
02/01/24

09/01/24

5. Develop a java program to create a class Bank that maintains two kinds of accounts for its customers, one called savings account and the other current account. The savings account provides compound interest & withdrawal facilities. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance & if the balance falls below this level, a service charge is imposed.

```

import java.util.*;
class Account {
    String customer_name;
    int account_number;
    char type;
    double balance;
    Account (String name, int num, char type) {
        this.customer_name = name;
        this.account_number = num;
        this.type = type;
        this.balance = 0;
    }
    void deposit (double amt) {
        this.balance += amt;
        System.out.println ("Amount " + amt + " deposited!");
    }
    void display () {
        System.out.println ("Balance is " + this.balance);
    }
    void withdraw (double amt) {
        if (this.balance >= amt) {
    
```

```
balance -= amt;  
System.out.println("Amount withdrawn");  
}  
else {  
    System.out.println("Insufficient funds");  
}  
}  
void interest(){  
    System.out.println("Interest not available  
for current account");  
}  
}  
  
class Current extends Account{  
    int min_balance = 500;  
    int penalty = 50;  
    Current(String name, int num){  
        super(name, num, 'c');  
    }  
    void withdraw(int amt){  
        if(this.balance > amt){  
            balance -= amt;  
            System.out.println("Amount withdrawn");  
        }  
        else {  
            System.out.println("Insufficient funds");  
        }  
        if(this.balance < min_balance){  
            balance -= penalty;  
            System.out.println("Balance below min");  
            System.out.println("Service charge imposed");  
        }  
    }  
}
```

class Savings extends Account {

Savings ( String name, int num ) {  
super ( name, num, 's' );

{

void interest () {

Scanner sc = new Scanner ( System. in );  
double rate = 6;

double time ;

System.out.println (" Enter time for interest ");  
time = sc.nextDouble();double new\_bal = this.balance \* ( Math. pow  
( ( 1 + ( rate / 100 ) ), time ) );System.out.println (" Your new balance  
will be " + new\_bal );

{

public class Main {

public static void main ( String [ ] args ) {

Scanner sc = new Scanner ( System. in );

System.out.println (" Enter customer name ");  
String name = sc.nextLine();System.out.println (" Enter account number ");  
int number = sc.nextInt();System.out.println (" Enter account type  
( s for Savings / c for Current ) ");

char type = sc.next().charAt ( 0 );

```
Account account;
if (type == 's'){
    account = new Savings(name, number);
} else if (type == 'c'){
    account = new Current(name, number);
} else {
    System.out.println("Invalid account type");
    return;
}

System.out.println("Account created successfully");
while (true) {
    int choice;
    System.out.println("Choose an operation:");
    System.out.println("1. Deposit");
    System.out.println("2. Display balance");
    System.out.println("3. Compute interest");
    System.out.println("4. Withdraw");
    System.out.println("Enter your choice");
    choice = sc.nextInt();
    switch (choice) {
        case 1:
            System.out.println("Enter deposit amount");
            double amt = sc.nextDouble();
            account.deposit(amt);
            break;
        case 2:
            account.display();
            break;
        case 3:
            account.interest();
            break;
    }
}
```

case 4:

System.out.println("Enter withdrawal amount");

double withdrawal\_amount = sc.nextDouble();  
account.withdraw(withdrawal\_amount);  
break;

Default:

System.out.println("Program exited");  
return;

3

Off.

Enter customer name:

Pranav - Anantha - Rao

Enter account number:

100035

Enter account type: (s for Savings / c for Current):

s

Account created successfully

Choose an operation:

1. Deposit
2. Display balance
3. Compute interest
4. Withdraw
5. Exit

Enter your choice: 1

Enter deposit amount: 1000

Enter your choice : 2

Balance is 1000.0

Enter your choice : 3

Enter time for interest : 3

Your new balance will be 1391.016

Enter your choice : 4

Enter withdrawal account : 100

Amount withdrawn

~~Enter your choice : 5~~

~~Program exited.~~

10/10/21

23/01/24

7. Create a package CIF which has two classes Student & Internals. The class Student has members like USN, name, sem. The class Internals derived from Student has an array that stores the Internals marks scored in five courses of the current semester of the student. Create another package SEF which has the class External which is a derived class of Student.

student.java:

```
package CIF;
import java.util.Scanner;
public class Student {
    protected String usn;
    protected String name;
    protected int sem;
    public void InputStudentDetails() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter USN: ");
        String usn = sc.next();
        System.out.println("Enter name: ");
        String name = sc.next();
        System.out.println("Enter sem: ");
        int sem = sc.nextInt();
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }
    public void displayStudentDetails() {
        System.out.println("USN: " + this.usn);
```

```
System.out.println("Name: " + this.name);  
System.out.println("Sem: " + this.sem);  
y
```

### internals.java:

```
package CIE;  
import java.util.Scanner;  
import CIE.student;  
public class internals extends student{  
    protected int marks[] = new int[5];  
    public void InputCTEMarks(){  
        Scanner sc = new Scanner(System.in);  
        for(int i = 0; i < 5; i++){  
            System.out.println("Enter marks of subject"  
                + (i+1) + ":");  
            marks[i] = sc.nextInt();  
        }  
    }  
}
```

### externals.java:

```
package SEE;  
import CIE.internals;  
import java.util.Scanner;  
public class externals extends internals{  
    protected int marks[];  
    protected int finalMarks[];
```

```
public external() {
    marks = new int[5];
    finalMarks = new int[5];
}

public void InputSEEMarks() {
    Scanner sc = new Scanner(System.in);
    for (int i = 0; i < 5; i++) {
        System.out.println("Enter marks of Subject " + (i + 1) + ":");
        marks[i] = sc.nextInt();
    }
}

public void calculateFinalMarks() {
    for (int i = 0; i < 5; i++) {
        finalMarks[i] = marks[i] / 2 + super.marks[i];
    }
}

public void displayFinalMarks() {
    displayStudentDetails();
    for (int i = 0; i < 5; i++) {
        finalMarks[i] =
            System.out.println("Subject " + (i + 1) + ":" +
                finalMarks[i]);
    }
}
```

main.java :

```
import 3 SEE.Externals;
class Math {
    public static void main (String args[])
    {
        int numofStudents = 2;
        Externals finalMarks [] = new Externals [numofStudents];
        for (int i = 0; i < numofStudents; i++)
        {
            finalMarks [i] = new Externals ();
            finalMarks [i].InputStudentDetails ();
            System.out.println ("Enter CIE marks : ");
            finalMarks [i].InputCIEMarks ();
            System.out.println ("Enter SEE marks : ");
            finalMarks [i].InputSEEMarks ();

            System.out.println ("Displaying data : \n");
            for (int i = 0; i < numofStudents; i++)
            {
                finalMarks [i].calculateFinalMarks ();
                finalMarks [i].displayFinalMarks ();
            }
        }
    }
}
```

O/P:

Enter USN :

1BM22CS201

Enter Name :

Pranav

Enter sem :

3

Enter CIE marks

Enter Marks of Subject:

45

Enter Marks of Subject 2:

46

3:

47

4:

49

5:

48

Enter SEE Marks:

Enter marks of Subject 1: 92

2: 85

3: 76

4: 89

5: 82

Enter USN:

1BM22CS200

Enter name:

Prajwal

Enter sem:

3

Enter CIF Marks

Enter Marks of Subject 1: 49

2: 49

3: 48

4: 42

5: 49

Enter SEE marks

Enter marks of subject 1: 89

2: 78

3: 99

4: 92

5: 85

### Displaying data:

USN: IBM22CS201

Name: Pranav

Sem: 3

Subject 1: 91

2: 88

3: 85

4: 93

5: 89

USN: IBM22CS200

Name: Prajwal

Sem: 3

Subject 1: 93

2: 88

3: 97

4: 88

5: 91

23/10/24

30/01/2014

CLASSMATE

Date \_\_\_\_\_

Page \_\_\_\_\_

Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and derived class called "Son" which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge() when the input age < 0. In Son class, implement a constructor which takes the ages and throws the exception when son's age  $\geq$  father's age.

```
import java.util.Scanner;
```

```
class WrongAge extends Exception{
```

```
    WrongAge(){
```

```
        super("Age error");
```

```
}
```

~~```
    WrongAge(String message){
```~~~~```
        super(message);
```~~~~```
}
```~~

```
class Father{
```

```
    private int age;
```

```
    public Father(int age) throws WrongAge{
```

```
        if(age < 0){
```

~~```
            throw new WrongAge("Age cannot be negative");
```~~

```
}
```

~~```
        this.age = age;
```~~~~```
}
```~~

```
int getAge() {  
    return this.age;
```

```
}
```

```
class Son extends Father {  
    private int age;
```

```
    public Son(int fAge, int sAge) throws  
        WrongAge {
```

```
        super(fAge);
```

```
        if (sAge <= fAge) {
```

```
            throw new WrongAge ("Son's age can't be  
            greater than father's age");
```

```
}
```

```
    if (sAge < 0) {
```

```
        throw new WrongAge ("Age cannot be negative");
```

```
}
```

```
    this.age = sAge;
```

```
}
```

```
int getSonAge() {
```

```
    return this.age;
```

```
}
```

```
public class Aged
```

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

```
    Scanner sc = new Scanner (System.in);
```

```
    try {
```

```
        System.out.println ("Enter Father's age");
```

```
int FAge = sc.nextInt();
System.out.println ("Enter Son's age : ");
int SAge = sc.nextInt();
```

```
Father fat = new Father (fage);
```

```
Son son = new Son (Sage, fage, Sage);
```

```
System.out.println ("father's age is "+ fat.getAge());
```

```
System.out.println ("Son's age is "+ son.getSonAge());
```

```
} catch (WrongAge e){
```

```
System.out.println ("Error: "+ e.getMessage());
```

```
sc.close();
```

```
}
```

Output :

Enter Father's age: 45

Enter Son's age: 20

Father's age: 45

Son's age: 20

Enter Father's age: 18

Enter Son's age: 20

Error: Son's age can't be greater than Father's age.

Enter Father's age: 20

Enter Son's age: -2

Age can't be negative.

Oct 02/24

8. WAP which creates two threads, one thread displaying "BMS College of Engineering" once every ten seconds and another displaying "CSE" once every two seconds

```
class m1 extends Thread {
```

```
    public void run() {
```

```
        while (true) {
```

```
            System.out.println("BMS College of Engineering");
```

```
            try {
```

```
                Thread.sleep(10000);
```

```
            }
```

```
            catch (InterruptedException e) {
```

```
                System.out.println("Interrupted;" + e);
```

```
}
```

```
}
```

```
}
```

```
class m2 extends Thread {
```

```
    public void run() {
```

```
        while (true) {
```

```
            System.out.println("CSE");
```

```
            try {
```

```
                Thread.sleep(2000);
```

```
            }
```

```
            catch (InterruptedException e) {
```

```
                System.out.println("Interrupted;" + e);
```

```
}
```

```
}
```

```
}
```

```
public class Main{  
    public static void main (String args[]){  
        m1 bms = new m1();  
        m2 cse = new m2();  
        bms.start();  
        cse.start();  
    }  
}
```

Output:

BMS College of Engineering

CSE

CSE

CSE

CSE

CSE

BMS College of Engineering

CSE

CSE

CSF

CSF

CSE

## 10. Demonstrate Inter-process Communication &amp; deadlock.

```
class Q {
```

```
    int n;
```

```
    boolean valueSet = false;
```

```
    synchronized int get() {
```

```
        while (!valueSet)
```

```
            try {
```

```
                System.out.println("In Consumer waiting");
```

```
                wait();
```

```
}
```

```
        catch (InterruptedException e) {
```

~~```
            System.out.println("Interrupted: " + e);
```~~

```
}
```

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

```
        valueSet = true;
```

```
        System.out.println("In Intimate Producer In");
```

```
        notify();
```

```
        return n;
```

```
}
```

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

```
    while (valueSet)
```

```
        try {
```

~~```
            System.out.println("In Producer waiting In");
```~~~~```
            wait();
```~~

```
}
```

```
    catch (InterruptedException e) {
```

~~```
        System.out.println("Interrupted: " + e);
```~~

```
}
```

```
this.n = n;  
valueset = true;  
System.out.println("Put: " + n);  
System.out.println("In Intimate Consumer");  
notify();  
}  
}
```

```
class Producer implements Runnable {
```

```
{ 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;
```

```
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 Main {
```

```
    public static void main (String args []) {  
        Q q = new Q ();  
        new Producer (q);  
        new Consumer (q);  
        System.out.println ("Press (ctrl+C to stop");  
    }  
}
```

Output:

Press Ctrl+C to stop

Put: 0

Intimate Consumer

Producer waiting

Got: 0

:

Put: 14

Intimate consumer

Got: 14

Consumed: 14

## DEADLOCK:

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 " + e);

}

~~System.out.println(name + " trying to call B.last()");~~

b.last();

}

void last () {

System.out.println("Inside A.last");

}

4

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 Main implements Runnable{
    A a = new A();
    B b = new B();
    Main(){
        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 other thread");
    }
}
  
```

O/P:

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.

- 10 Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the result field when the divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were zero, the program would throw an ArithmeticException. Display the exception in a message dialog box.

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

class SwingDemo {
    SwingDemo() {
        JFrame jfrm = new JFrame ("Divider App");
        jfrm.setSize (275, 150);
        jfrm.setLayout (new FlowLayout ());
        jfrm.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);
        JLabel jlab = new JLabel ("Enter the divisor  
and dividend : ");
        JTextField aijf = new JTextField (8);
        JTextField bjjf = new JTextField (8);
        JButton button = new JButton ("Button");
        JButton calculate = new JButton ("Calculate");
        JLabel err = new JLabel ();
        JLabel alab = new JLabel ();
        JLabel blab = new JLabel ();
        JLabel ansLab = new JLabel ();
        jfrm.add (err);
    }
}
```

```

jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);

```

```

ActionListener i = new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        System.out.println("Action event from a
                           text field");
    }
};

```

```

ajtf.addActionListener(i);
bjtf.addActionListener(i);

```

```

button.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        try {
            int a = Integer.parseInt(ajtf.getText());
            int b = Integer.parseInt(bjtf.getText());
            int ans = a/b;
            alab.setText("In A = " + a);
            blab.setText("In B = " + b);
            anslab.setText("In Ans = " + ans);
        }
    }
});

```

```

catch (NumberFormatException e) {
    alab.setText(" ");
    blab.setText(" ");
    anslab.setText(" ");
}

```

```

err. setText("Enter only Integers!");
}
catch (ArithmeticException e){
    alab.setText("");
    blab.setText("");
    anslab.setText("");
    err.setText("B should be NON zero!");
}
}
}
}

frm.setVisible(true);
}

public static void main(String args[]){
    SwingUtilities.invokeLater(new Runnable(){
        public void run(){
            new SwingDemo();
        }
    });
}
}

```

O/P:

Divider App

Enter the divider and dividend:

|     |   |
|-----|---|
| 200 | 2 |
|-----|---|

(Calculate)  $A = 200 \quad B = 2 \quad \text{Ans} = 100$

20.02.24

## Lab 1

**Develop a Java program that prints all real solutions to the quadratic equation  $ax^2 + bx + c = 0$ . Read in a, b, c and use the quadratic formula. If the discriminant  $b^2 - 4ac$  is negative, display a message stating that there are no real solutions.**

```
import java.util.Scanner;

class Quadratic
{
    int a, b, c;
    double r1, r2, d;

    void getd()
    {
        Scanner s = new Scanner(System.in); System.out.println("Enter the coefficients of a,b,c"); a =
            s.nextInt();
        b = s.nextInt(); c = s.nextInt();

    }

    void compute()
    {
        while(a==0)
        {
            System.out.println("Not a quadratic equation"); System.out.println("Enter a non zero value
                for a:"); Scanner s = new Scanner(System.in);
            a = s.nextInt();
        }

        d = b*b-4*a*c;

        if(d==0)
```

```

{
    r1 = (-b)/(2*a);
    System.out.println("Roots are real and equal"); System.out.println("Root1 = Root2 = " +
    r1);
}
else if(d>0)
{
    r1 = ((-b)+(Math.sqrt(d)))/(double)(2*a);
    r2 = ((-b)-(Math.sqrt(d)))/(double)(2*a); System.out.println("Roots are real and distinct");
    System.out.println("Root1 = " + r1 + " Root2 = " + r2);
}
else if(d<0)
{
    System.out.println("Roots are imaginary"); r1 = (-b)/(2*a);
    r2 = Math.sqrt(-d)/(2*a); System.out.println("Root1 = " + r1 + " + i" + r2);
    System.out.println("Root1 = " + r1 + " - i" + r2);
}
}

class QuadraticMain
{
    public static void main(String args[])
    {
        Quadratic q = new Quadratic(); q.getd();
        q.compute();
    }
}

```

```
    }  
}  
}
```

## Lab 2

**Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.**

```
import java.util.Scanner;  
class Subject  
{  
    int sub_marks;  
    int credits;  
    int grade;  
}  
  
class SGPA  
{  
    Subject subject[];  
    String USN,name;  
    double sgpa;  
    Scanner sc = new Scanner(System.in);  
  
    SGPA()  
    {  
        subject=new Subject[8];  
        for(int i=0;i<8;i++)  
        {  
            subject[i]=new Subject();  
        }  
    }  
  
    void getData()  
    {  
        System.out.println("Enter name of Student.");  
        name=sc.nextLine();  
        System.out.println("Enter USN of Student.");  
        USN=sc.nextLine();  
    }  
}
```

```

void getMarks()
{
    for(int i=0;i<8;i++)
    {
        System.out.println("Enter subject "+(i+1)+" marks");
        subject[i].sub_marks=sc.nextInt();
        System.out.println("Enter subject "+(i+1)+" credits");
        subject[i].credits=sc.nextInt();
    }

    for(int i=0;i<8;i++)
    {
        if(subject[i].sub_marks>=90 && subject[i].sub_marks<=100)
        {
            subject[i].grade=10;
        }
        else if(subject[i].sub_marks>=80 && subject[i].sub_marks<90)
        {
            subject[i].grade=9;
        }
        else if(subject[i].sub_marks>=70 && subject[i].sub_marks<80)
        {
            subject[i].grade=8;
        }
        else if(subject[i].sub_marks>=60 && subject[i].sub_marks<70)
        {
            subject[i].grade=7;
        }
        else if(subject[i].sub_marks>=50 && subject[i].sub_marks<60)
        {
            subject[i].grade=6;
        }
        else if(subject[i].sub_marks>=40 && subject[i].sub_marks<50)
        {
            subject[i].grade=5;
        }
        else if(subject[i].sub_marks>=0 && subject[i].sub_marks<40)
        {
            subject[i].grade=0;
        }
        else
        {
            System.out.println("Enter valid score");
            break;
        }
    }
}

```

```
        }
    }

double compute()
{
    double sum=0.0,num;
    for(int i=0;i<8;i++)
    {
        num=(subject[i].grade)*(subject[i].credits);
        sum+=num;
    }
    sgpa=sum/20;
    return sgpa;
}

void display(double sgpa)
{
    System.out.println("Name of candidate: "+name);
    System.out.println("USN of candidate: "+USN);
    System.out.println("SGPA= "+sgpa);
}

class Main
{
    public static void main(String args[])
    {
        System.out.println("Prajwal P\n1BM22CS200\n\n");
        SGPA ob=new SGPA();
        ob.getData();
        ob.getMarks();
        double a=ob.compute();
        ob.display(a);
    }
}
```

## Lab 3

**Create a class Book which contains four members: name, author, price, num\_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a `toString()` method that could display the complete details of the book. Develop a Java program to create n book objects.**

```
import java.util.Scanner;  
class Book  
{  
    String name,author;  
    int price;  
    int num_pages;  
  
    Book(String name,String author,int price,int num_pages){  
        this.name=name;  
        this.author=author;  
        this.price=price;  
        this.num_pages=num_pages;  
    }  
  
    public String toString(){  
        String n,a,p,N;  
        n="\n"+ "Name of Book: "+name+"\n";  
        a="Author of Book: "+author+"\n";  
        p="Price of Book: "+price+"\n";  
        N="Number of pages: "+num_pages+"\n";  
        return n+a+p+N;  
    }  
}
```

```
class Main
{
    public static void main(String args[]){
        Scanner sc=new Scanner(System.in);
        System.out.println("\nEnter number of books: ");
        int n=sc.nextInt();
        Book b[]=new Book[n];
        String name,author;
        int price,num;
        sc.nextLine();
        for(int i=0;i<n;i++){
            System.out.println("Enter name of book: ");
            name=sc.nextLine();
            System.out.println("Enter author's name: ");
            author=sc.nextLine();
            System.out.println("Enter price: ");
            price=sc.nextInt();
            System.out.println("Enter number of pages: ");
            num=sc.nextInt();
            b[i]= new Book(name,author,price,num);
        }
        System.out.println("Book Details: ");
        for(int i=0;i<n;i++){
            System.out.println(b[i].toString());
        }
    }
}
```

## Lab 4

**Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea( ). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the classShape. Each one of the classes contain only the method printArea( ) that prints the area of the given shape.**

```
import java.util.Scanner;

abstract class Shape{
    double dim1,dim2;
    Shape(double a,double b){
        dim1=a;
        dim2=b;
    }
    abstract void printArea();
}

class Rectangle extends Shape{
    Rectangle(double a,double b){
        super(a,b);
    }
    void printArea(){
        if(dim1<0 || dim2<0){
            System.out.println("Invalid input for Rectangle");
        }
        else{
            System.out.println("\nArea of Rectangle is "+(dim1*dim2));
        }
    }
}

class Triangle extends Shape{
    Triangle(double a, double b){
        super(a,b);
    }
    void printArea(){
        if(dim1<0 || dim2<0){
            System.out.println("Invalid input for Triangle");
        }
        else{
            System.out.println("Area of Triangle is "+(dim1*dim2)/2);
        }
    }
}
```

```

        }

}

class Circle extends Shape{
    Circle(double a, double b){
        super(a,b);
    }
    final double pi=3.14159;
    void printArea(){
        if(dim1<0 || dim2<0){
            System.out.println("Invalid input for Circle");
        }
        else{
            System.out.println("Area of Circle is "+pi*Math.pow(dim1,2));
        }
    }
}

class Main{
    public static void main(String args[]){

        Scanner sc=new Scanner(System.in);
        double a,b;

        System.out.println("Enter sides of rectangle: ");
        a=sc.nextDouble();
        b=sc.nextDouble();
        Rectangle r=new Rectangle(a,b);

        System.out.println("Enter height and base length of Triangle: ");
        a=sc.nextDouble();
        b=sc.nextDouble();
        Triangle t=new Triangle(a,b);

        System.out.println("Enter radius of Circle: ");
        a=sc.nextDouble();
        Circle c=new Circle(a,1);

        Shape s;
        s=r;
        s.printArea();
        s=t;
        s.printArea();
        s=c;
        s.printArea();
    }
}

```

## Lab 5

**Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Create a class Account that stores customer name, account number and type of account. From this derive the classes Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:**

- a) Accept deposit from customer and update the balance.
- b) Display the balance.
- c) Compute and deposit interest
- d) Permit withdrawal and update the balance

**Check for the minimum balance, impose penalty if necessary and update the balance.**

```
import java.util.*;  
class Account{  
    String customer_name;  
    int account_number;  
    char type;  
    double balance;  
    Account(String name, int num, char type){  
        this.customer_name = name;  
        this.account_number = num;  
        this.type = type;  
        this.balance = 0;  
    }  
    void deposit(double amt){  
        this.balance += amt;  
        System.out.println("Amount " + amt + " deposited!");  
    }
```

```
}

void display(){
    System.out.println("Balance is " + this.balance);
}

void withdraw(double amt){
    if(this.balance >= amt){
        balance -= amt;
        System.out.println("Amount withdrawn");
    }
    else{
        System.out.println("Insufficient funds");
    }
}

void interest(){
    System.out.println("Interest not available for current account");
}
```

```
class Current extends Account{
    int min_balance = 500;
    int penalty = 50;
    Current(String name, int num){
        super(name, num, 'c');
    }
    void withdraw(int amt){
        if(this.balance > amt){
            balance -= amt;
            System.out.println("Amount withdrawn");
        }
        else{
```

```

        System.out.println("Insufficient funds");
    }
    if(this.balance < min_balance){
        balance -= penalty;
        System.out.println("Balance below minimum");
        System.out.println("Service charge imposed");
    }
}

class Savings extends Account{
    Savings(String name, int num){
        super(name, num, 's');
    }
    void interest(){
        Scanner sc = new Scanner(System.in);
        double rate = 6;
        double time;
        System.out.println("Enter time for interest: ");
        time = sc.nextDouble();
        double new_bal = this.balance*(Math.pow((1+(rate/100)), time));
        System.out.println("Your new balance will be " + new_bal);
    }
}

public class Main
{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter customer name: ");

```

```
String name = sc.nextLine();
System.out.println("Enter account number: ");
int number = sc.nextInt();
System.out.println("Enter account type (s for Savings/c for Current): ");
char type = sc.next().charAt(0);
Account account;
if (type == 's') {
    account = new Savings(name, number);
} else if (type == 'c') {
    account = new Current(name, number);
} else {
    System.out.println("Invalid account type.");
    return;
}
System.out.println("Account created successfully");
while(true){
    int choice;
    System.out.println("Choose an operation:");
    System.out.println("1. Deposit");
    System.out.println("2. Display balance");
    System.out.println("3. Compute and deposit interest");
    System.out.println("4. Withdraw");
    System.out.println("5. Exit");
    System.out.print("Enter your choice: ");
    choice = sc.nextInt();
    switch(choice) {
        case 1:
            System.out.print("Enter deposit amount: ");
            double amt = sc.nextDouble();
            account.deposit(amt);
```

```
        break;

    case 2:
        account.display();
        break;

    case 3:
        account.interest();
        break;

    case 4:
        System.out.print("Enter withdrawal amount: ");
        double withdrawAmount = sc.nextDouble();
        account.withdraw(withdrawAmount);
        break;

    default:
        System.out.print("Program exited");
        return;
    }
}

}
```

## Lab 6

**Create a package CIE which has two classes- Student and Internals. The class Personal has members like usn, name, sem. The class internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External which is a derived class of Student. This class has an array that stores the SEE marks scored in five courses of the current semester of the student. Import the two packages in a file that declares the final marks of n students in all five courses.**

student.java:

```
package CIE;
import java.util.Scanner;
public class student{
    protected String usn;
    protected String name;
    protected int sem;
    public void InputStudentDetails(){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter USN: ");
        String usn = sc.next();
        System.out.println("Enter name: ");
        String name = sc.next();
        System.out.println("Enter sem: ");
        int sem = sc.nextInt();
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }
    public void displayStudentDetails(){
        System.out.println("USN: " + this.usn);
        System.out.println("Name: " + this.name);
        System.out.println("Sem: " + this.sem);
    }
}
```

internals.java:

```
package CIE;
import java.util.Scanner;
import CIE.student;
```

```

public class internals extends student{
    protected int marks[] = new int[5];
    public void InputCIEMarks(){
        Scanner sc = new Scanner(System.in);
        for(int i = 0; i < 5; i++){
            System.out.println("Enter Marks of Subject " + (i+1) + ":");
            marks[i] = sc.nextInt();
        }
    }
}

```

externals.java:

```

package SEE;
import CIE.internals;
import java.util.Scanner;
public class externals extends internals {
    protected int marks[];
    protected int finalMarks[];
    public externals() {
        marks = new int[5];
        finalMarks = new int[5];
    }
    public void inputSEEmarks() {
        Scanner sc = new Scanner(System.in);
        for(int i=0;i<5;i++) {
            System.out.print("Enter marks of Subject "+(i+1)+" : ");
            marks[i] = sc.nextInt();
        }
    }
    public void calculateFinalMarks() {
        for(int i=0;i<5;i++){
            finalMarks[i] = marks[i]/2 + super.marks[i];
        }
    }
    public void displayFinalMarks() {
        displayStudentDetails();
        for(int i=0;i<5;i++){
            System.out.println("Subject " + (i+1) + ": " + finalMarks[i]);
        }
    }
}

```

main.java:

```
import SEE.externals;
class Main {
    public static void main(String args[]){
        int numOfStudents = 2;
        externals finalMarks[] = new externals[numOfStudents];
        for(int i=0;i<numOfStudents;i++){
            finalMarks[i] = new externals();
            finalMarks[i].InputStudentDetails();
            System.out.println("Enter CIE marks");
            finalMarks[i].InputCIEMarks();
            System.out.println("Enter SEE marks");
            finalMarks[i].inputSEEmarks();
        }
        System.out.println("Displaying data:\n");
        for(int i=0;i<numOfStudents;i++){
            finalMarks[i].calculateFinalMarks();
            finalMarks[i].displayFinalMarks();
        }
    }
}
```

## Lab 7

**Write a program that demonstrates handling of exceptions in inheritance tree.**

**Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge() when the input age<0. In Son class, implement a constructor that cases both father and son’s age and throws an exception if son’s age is >=father’s age.**

```
import java.util.Scanner;

class WrongAge extends Exception{
    WrongAge(){
        super("Age error");
    }
    WrongAge(String message){
        super(message);
    }
}
```

```

}

class Father{
    private int age;
    public Father(int age) throws WrongAge {
        if(age < 0){
            throw new WrongAge("Age cannot be negative");
        }
        this.age = age;
    }

    int getAge(){
        return this.age;
    }
}

class Son extends Father{
    private int age;

    public Son(int FAge, int SAge) throws WrongAge {
        super(FAge);
        if(SAge >= FAge){
            throw new WrongAge("Son's age can't be greater than Father's age");
        }
        if(SAge < 0){
            throw new WrongAge("Age cannot be negative");
        }
        this.age = SAge;
    }

    int getSonAge(){
        return this.age;
    }
}

public class Age{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        try{
            System.out.println("Enter Father's age");
            int Fage = sc.nextInt();
            System.out.println("Enter Son's age");
            int Sage = sc.nextInt();

            Father fat = new Father(Fage);
            Son son = new Son(Fage, Sage);
        }
    }
}

```

```

        System.out.println("Father's Age is " + fat.getAge());
        System.out.println("Son's Age is " + son.getSonAge());
    } catch (WrongAge e){
        System.out.println("Error: " + e.getMessage());
    }
    sc.close();
}
}

```

## Lab 8

**Write a program that creates two threads, one thread displaying “BMS College of Engineering” once every ten seconds and another displaying “CSE” once every two seconds.**

```

class m1 extends Thread{
    public void run(){
        while(true){
            System.out.println("BMS College of Engineering");
            try{
                Thread.sleep(10000);
            }
            catch(InterruptedException e){
                System.out.println("Interrupted: " + e);
            }
        }
    }
}

class m2 extends Thread{
    public void run(){
        while(true){
            System.out.println("CSE");
            try{
                Thread.sleep(2000);
            }
            catch(InterruptedException e){
                System.out.println("Interrupted: " + e);
            }
        }
    }
}

```

```

public class Main{
    public static void main(String args[]){
        m1 bms = new m1();
        m2 cse = new m2();
        bms.start();
        cse.start();
    }
}

```

## Lab 9

**Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were Zero, the program would throw an ArithmeticException Display the exception in a message dialog box.**

```

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

class SwingDemo{
    SwingDemo(){
        JFrame jfrm = new JFrame("Divider App");
        jfrm.setSize(275, 150);
        jfrm.setLayout(new FlowLayout());
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel jlab = new JLabel("Enter the divider and divident:");
        JTextField ajtf = new JTextField(8);
        JTextField bjtf = new JTextField(8);
        JButton button = new JButton("Calculate");
        JLabel err = new JLabel();
        JLabel alab = new JLabel();
        JLabel blab = new JLabel();
        JLabel anslab = new JLabel();
        jfrm.add(err);
        jfrm.add(jlab);
        jfrm.add(ajtf);
        jfrm.add(bjtf);
        jfrm.add(button);
        jfrm.add(alab);
        jfrm.add(blab);
    }
}

```

```

jfrm.add(anslab);

ActionListener l = new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        System.out.println("Action event from a text field");
    }
};

ajtf.addActionListener(l);
bjtf.addActionListener(l);

button.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        try{
            int a = Integer.parseInt(ajtf.getText());
            int b = Integer.parseInt(bjtf.getText());
            int ans = a/b;
            alab.setText("\nA = " + a);
            blab.setText("\nB = " + b);
            anslab.setText("\nAns = "+ ans);
        }
        catch(NumberFormatException e){
            alab.setText("");
            blab.setText("");
            anslab.setText("");
            err.setText("Enter Only Integers!");
        }
        catch(ArithmaticException e){
            alab.setText("");
            blab.setText("");
            anslab.setText("");
            err.setText("B should be NON zero!");
        }
    }
});

jfrm.setVisible(true);
}

public static void main(String args[]){
    SwingUtilities.invokeLater(new Runnable(){
        public void run(){
            new SwingDemo();
        }
    });
}

```

```

        });
    }
}

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("InterruptedExceptioncaught");
            }
        System.out.println("Got: " + n);
        valueSet = false;
        System.out.println("\nIntimate 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("\nIntimate Consumer\n");
        notify();
    }
}

class Producer implements Runnable {
    Q q;

```

## Lab 10

### Demonstrate Inter process Communication and deadlock. IPC

```

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 Main {
    public static void main(String args[]) {
        Q q = new Q();
        new Producer(q);
        new Consumer(q);
        System.out.println("Press Control-C to stop.");
    }
}

```

## **Deadlock**

```
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() {
        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();
    }

    void last() {
        System.out.println("Inside A.last");
    }
}
```

```
}

class Main implements Runnable {

    A a = new A();

    B b = new B();

    Main() {
        Thread.currentThread().setName("MainThread");

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

        t.start();

        a.foo(b); // get lock on a in this
        System.out.println("Back in main thread");
    }

    public void run() {
        b.bar(a); // get lock on b in other
        System.out.println("Back in other thread");
    }
}

public static void main(String args[]) {
    new Main();
}
```