VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

SHASHANK RAVINDRA(1BM23CS312)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019 Sep 2024-Jan 2025

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

<u>CERTIFICATE</u>

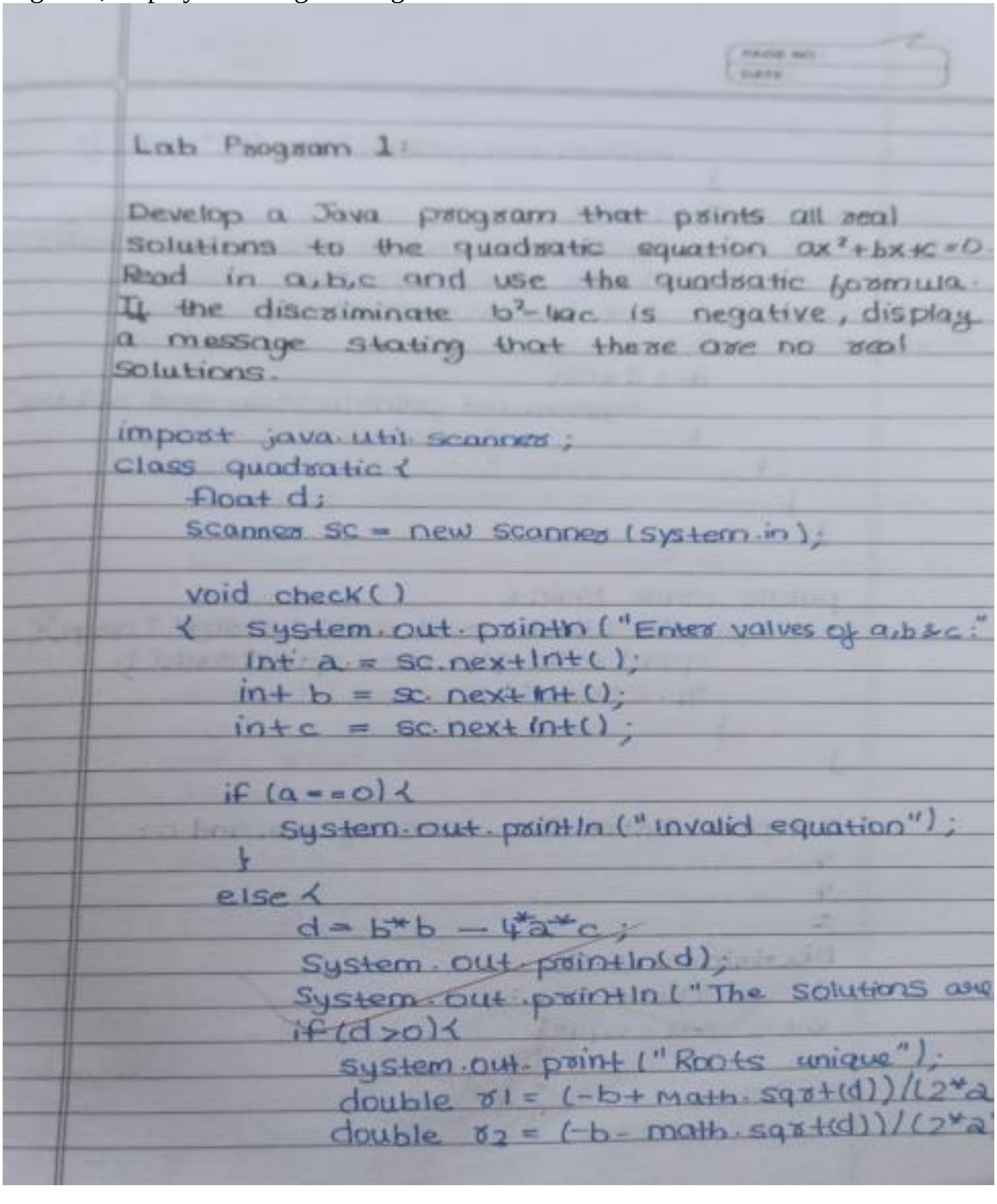
This is to certify that the Lab work entitled "**OBJECT ORIENTED JAVA PROGRAMMING**" carried out by **SHASHANK RAVINDRA KARANAM(1BM23CS312),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Object-Oriented Java Programming Lab - (23CS3PCOOJ)** work prescribed for the said degree.

Dr. Nandhini Vineeth	**Dr. Kavitha Sooda**
Associate Professor, Department of CSE, BMSCE, Bengaluru	Professor and Head, Department of CSE BMSCE, Bengaluru

INDEX

LABORATORY PROGRAM – 1

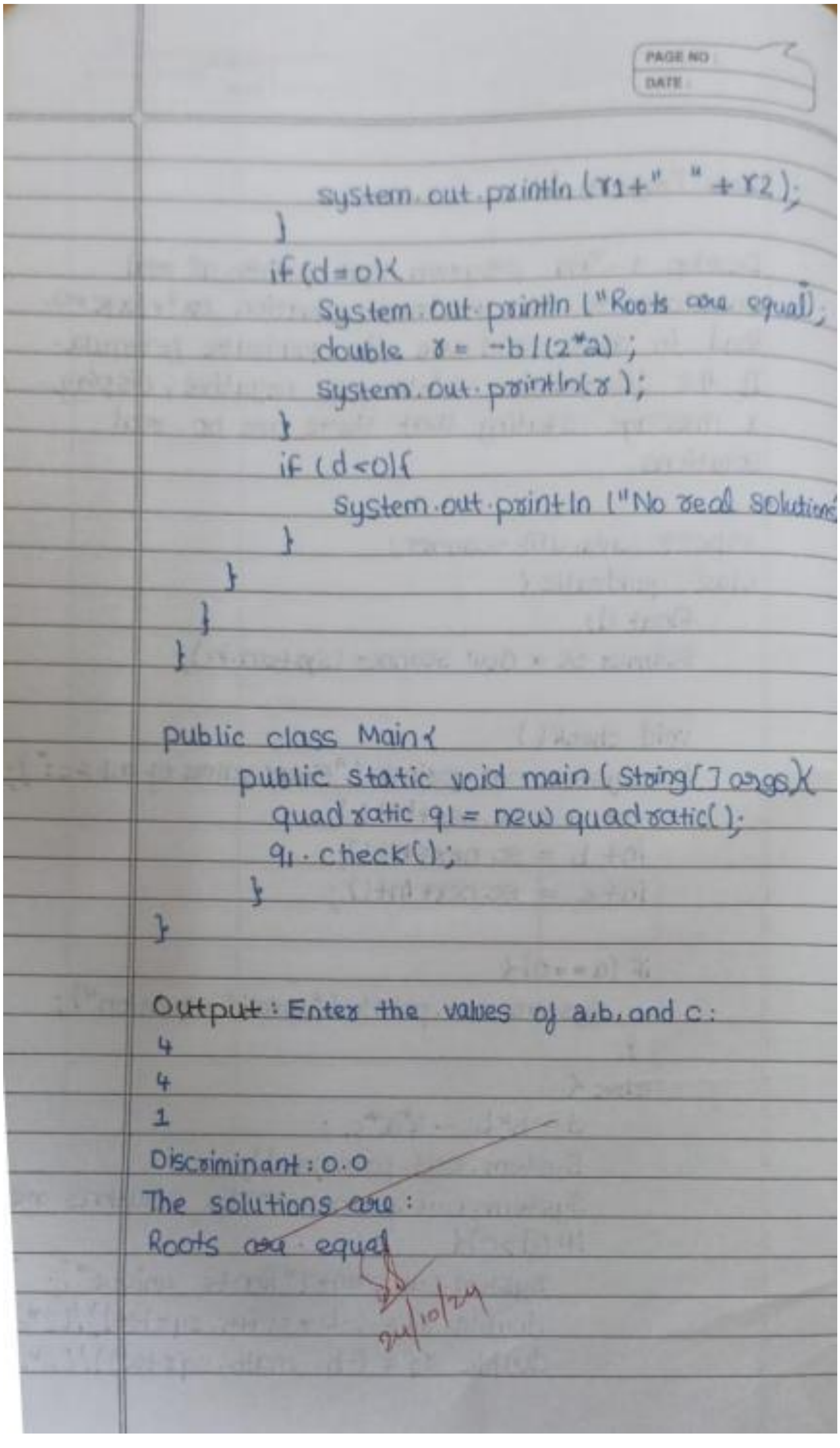
Develop a Java program that prints all real solutions to the quadratic equation ax2 +bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2 -4ac is negative, display a message stating that there are no real solutions.



Lab Program 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 discriminate b^2-4ac is negative, display a message stating that there are no real solutions.

```
import java.util.Scanner;
class quadratic {
      float d;
      Scanner sc = new Scanner (System.in);

      void check( )
      {   System.out.println("Enter values of a,b&c:"
          Int a = sc.nextInt( );
          int b = sc.nextInt();
          int c = sc.nextInt() ;

          if (a==0){
             System.out.println("invalid equation");
          }
          else{
                d= b*b - 4*a*c ;
                System.out.println(d);
                System.out.println("The solutions are
                if(d>0){
                   System.out.print("Roots unique");
                   double r1 = (-b+Math.sqrt(d))/(2*a
                   double r2 = (-b- math.sqrt(d))/(2*a)
```



PAGE NO :
DATE :

```
            system.out.println(r1+"  "+r2);
        }
        if(d=0){
            System.out.println("Roots are equal);
            double r = -b/(2*a);
            system.out.println(r);
        }
        if(d<0){
            System.out.println("No real solutions
        }
      }
    }
  }

public class Main{
    public static void main(String[] args){
        quadratic q1 = new quadratic();
        q1.check();
    }
}
```

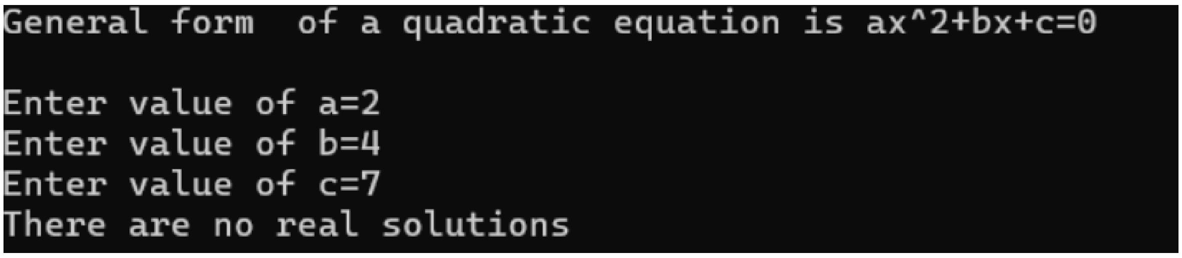
```
Output: Enter the values of a,b,and c:
4
4
1
Discriminant: 0.0
The solutions are :
Roots are equal
```

[illegible]
24/10/24

Soft copy of the program

```
import java.util.Scanner;
class Quad_Eq_cal{
 public static void main(String [] args){
 int y=0;
 Scanner sc=new Scanner(System.in);
 System.out.println("General form of a quadratic equation is ax^2+bx+c=0");
 do{
 System.out.print("\nEnter value of a=");
 int a=sc.nextInt();
 System.out.print("Enter value of b=");
 int b=sc.nextInt();
 System.out.print("Enter value of c=");
 int c=sc.nextInt();
 float d=(float)(Math.pow(b,2)-4*a*c);
 if(d<0){
 System.out.println("There are no real solutions");
 }
 else if(d==0){
 System.out.println("It has one repeated root(2 equal roots):");
 float r=-b/(2.0f*a);
 System.out.println("x="+r);
 }
 else{
 System.out.println("It has two distinct roots:");
 double r1=((-b+Math.sqrt(d))/(2*a));
 System.out.println("x1="+r1);
 double r2=((-b-Math.sqrt(d))/(2*a));
 System.out.println("x2="+r2);
 }
 System.out.println("\nDo you want to calculate again?(yes=0 and no=1): ");
 y=sc.nextInt();
 }while(y==0);
 }
}
```

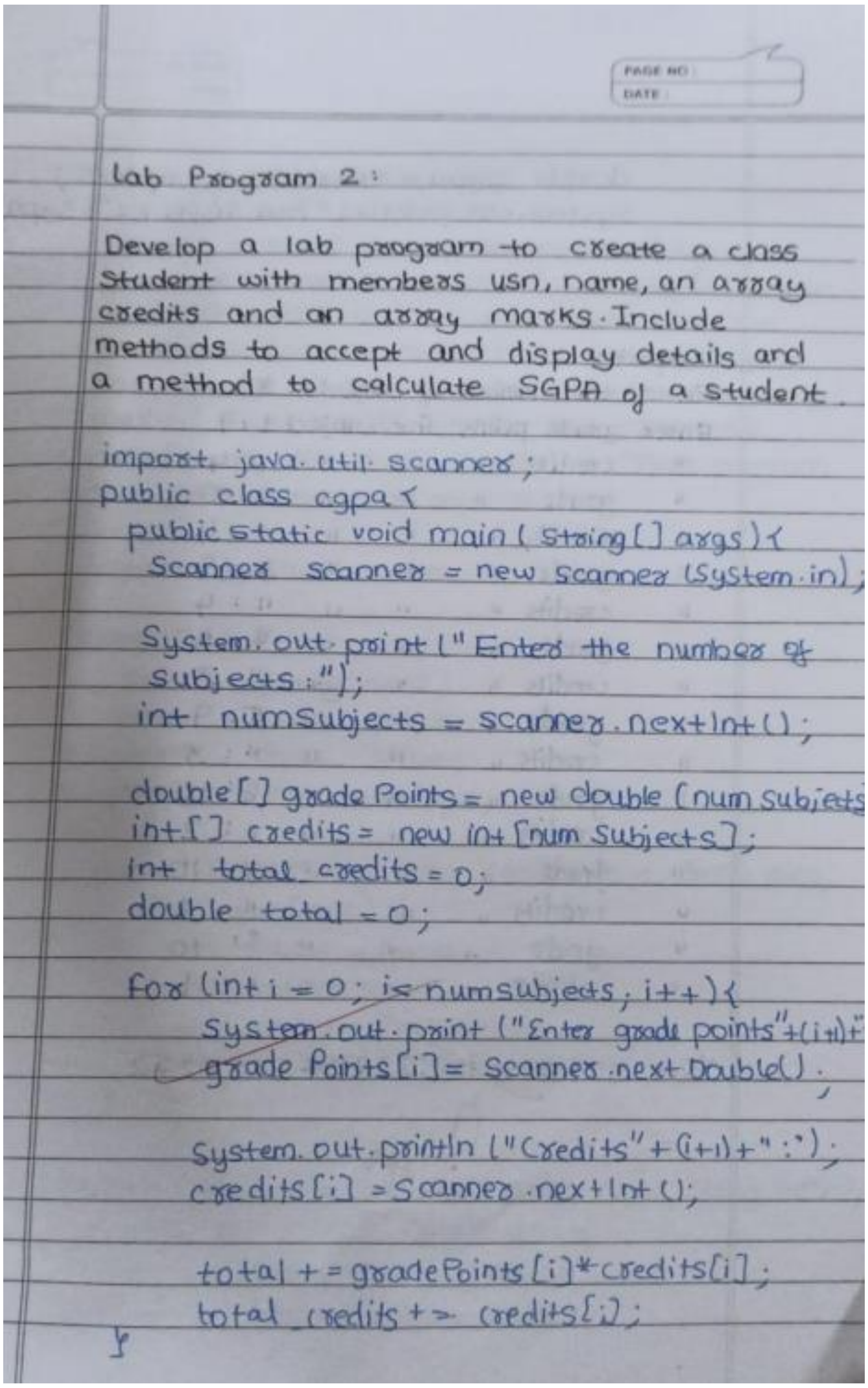
OUTPUT:



```
General form  of a quadratic equation is ax^2+bx+c=0

Enter value of a=2
Enter value of b=4
Enter value of c=7
There are no real solutions
```

LABORATORY PROGRAM – 2

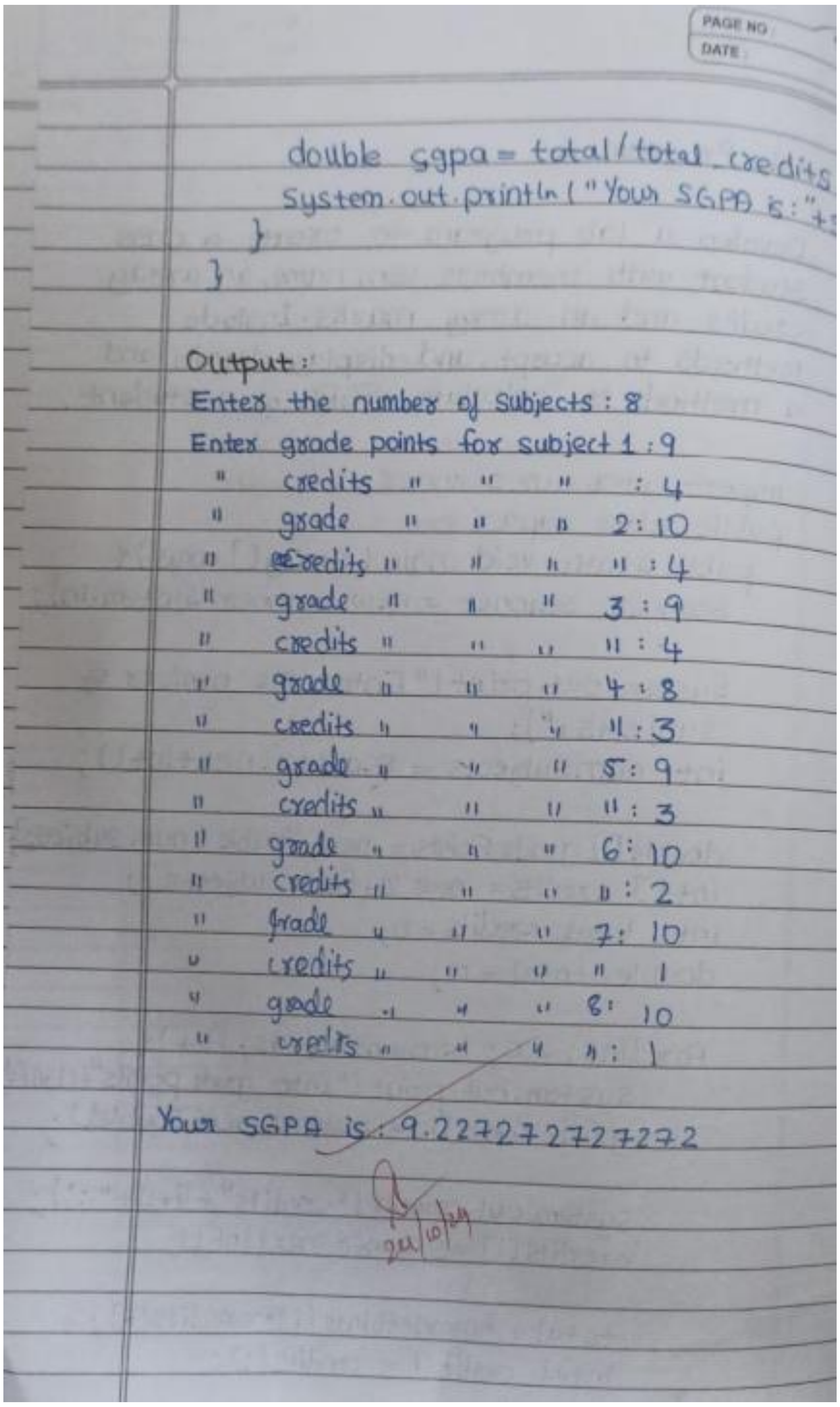
Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.



lab Program 2:

Develop a lab 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;
public class cgpa {
  public static void main(String[] args){
    Scanner scanner = new scanner(System.in);

   System.out.print("Enter the number of
    subjects:");
   int numSubjects = scanner.nextInt();

   double[] gradePoints = new double (num subjects
   int[] credits = new int[num Subjects];
   int total_credits = 0;
   double total = 0;

   for (int i = 0; i<numsubjects; i++){
        System.out.print("Enter grade points"+(i+1)+"
        gradePoints[i] = scanner.nextDouble();

        System.out.println("Credits"+(i+1)+":");
        credits[i] =scanner.nextInt();

        total += gradePoints[i]*credits[i];
        total_credits += credits[i];
   }
```



PAGE NO :
DATE :

```
        double sgpa = total/total_credits
        System.out.println("Your SGPA is:"+
    }
}
```

Output:

```
Enter the number of Subjects : 8
Enter grade points for subject 1 : 9
  "    credits    "     "     "  : 4
  "    grade      "     "     "  2 : 10
  "    ~~c~~credits  "  "     "  " : 4
  "    grade      "     "     "  3 : 9
  "    credits    "     "     "  " : 4
  "    grade      "     "     "  4 : 8
  "    credits    "     "     "  " : 3
  "    grade      "     "     "  5 : 9
  "    credits    "     "     "  " : 3
  "    grade      "     "     "  6 : 10
  "    credits    "     "     "  " : 2
  "    grade      "     "     "  7 : 10
  "    credits    "     "     "  "   1
  "    grade      "     "     "  8 : 10
  "    credits    "     "     "  " : 1

Your SGPA is : 9.2272727272727272
```

[illegible] 24/10/24

import java.util.Scanner;

class Subject {

 int subM;

 int cred;

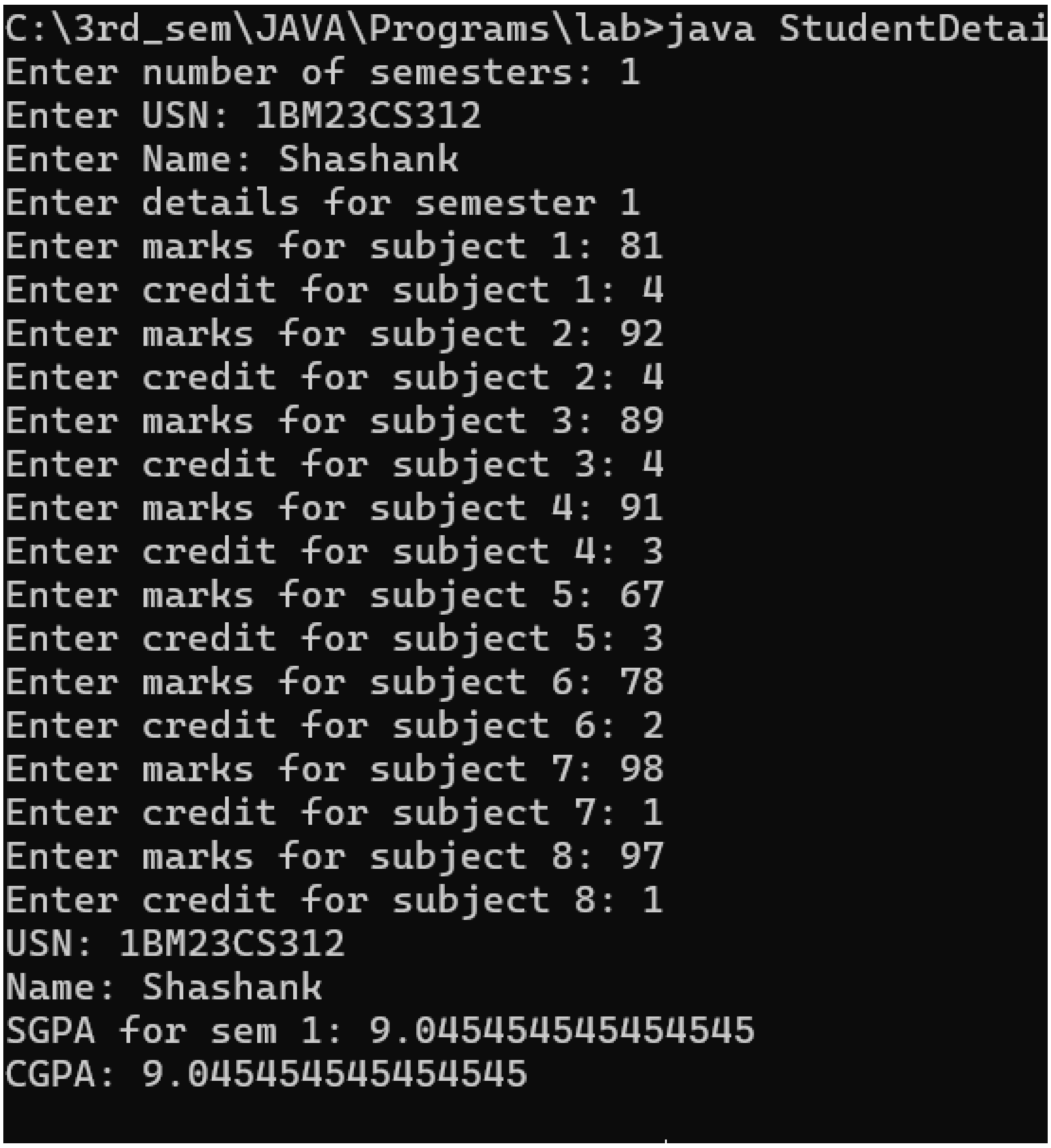
 int grade;

 void setSubDet(int marks, int cred) {

```
 this.subM = marks;
 this.cred = cred;
 if (subM >= 90) {
 grade = 10;
 } else if (subM >= 80) {
 grade = 9;
 } else if (subM >= 70) {
 grade = 8;
 } else if (subM >= 60) {
 grade = 7;
 } else if (subM >= 50) {
 grade = 6;
 } else if (subM >= 40) {
 grade = 5;
 } else {
 grade = 0;
 }
 }
}
class Student {
 Scanner s = new Scanner(System.in);
 Subject[] subjects = new Subject[8];
 Student() {
 for (int i = 0; i < subjects.length; i++) {
 subjects[i] = new Subject();
 }
```

```
}
void getMarks() {
for (int i = 0; i < subjects.length; i++) {
System.out.print("Enter marks for subject " + (i + 1) + ": ");
int marks = s.nextInt();
System.out.print("Enter credit for subject " + (i + 1) + ": ");
int cred = s.nextInt();
subjects[i].setSubDet(marks, cred);
}
}
double calSGPA() {
double Score = 0;
int totalCred = 0;
double SGPA = 0.0;
for (Subject subject : subjects) {
Score += (subject.grade * subject.cred);
totalCred += subject.cred;
}
if (totalCred > 0) {
SGPA = Score / totalCred;
} else {
SGPA = 0;
}
return SGPA;
}
}
```

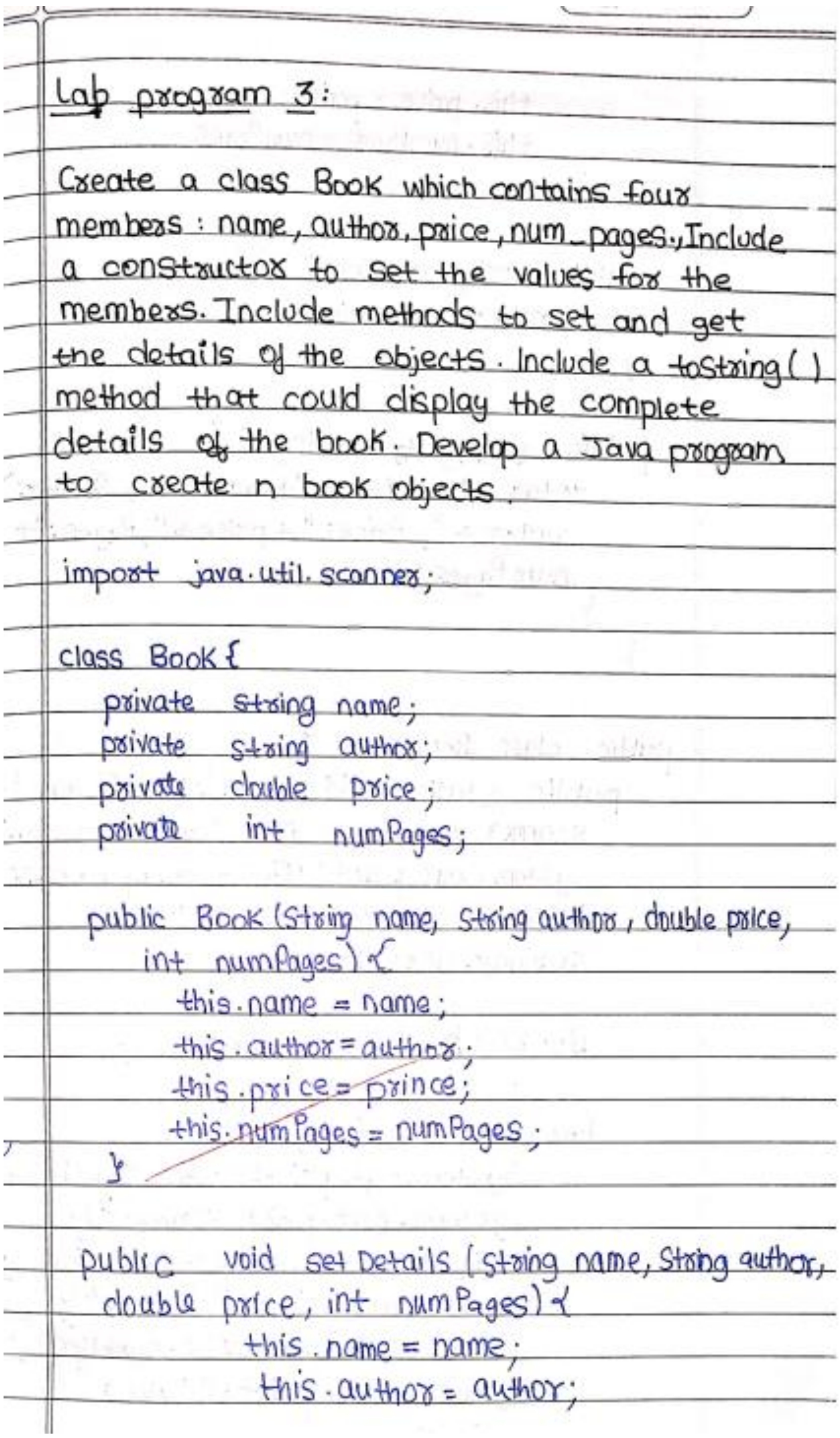
```
public class StudentDetails {
 public static void main(String[] arg) {
 Scanner sc = new Scanner(System.in);

 System.out.print("Enter number of semesters: ");
 int numSems = sc.nextInt();

 Student[] students = new Student[numSems];
 double cumulativeSGPA = 0.0;

 System.out.print("Enter USN: ");
 String usn = sc.next();

 System.out.print("Enter Name: ");
 String name = sc.next();

 for (int i = 0; i < numSems; i++) {
 System.out.println("Enter details for semester " + (i + 1));
 students[i] = new Student();
 students[i].getMarks();
 double semSGPA = students[i].calSGPA();
 cumulativeSGPA += semSGPA;
 }

 for (int i = 0; i < numSems; i++) {
 System.out.println("USN: " + usn);
System.out.println("Name: " + name);

System.out.println("SGPA for sem " + (i + 1) + ": " + students[i].calSGPA());

}


double CGPA = cumulativeSGPA / numSems;

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



```
C:\3rd_sem\JAVA\Programs\lab>java StudentDetai
Enter number of semesters: 1
Enter USN: 1BM23CS312
Enter Name: Shashank
Enter details for semester 1
Enter marks for subject 1: 81
Enter credit for subject 1: 4
Enter marks for subject 2: 92
Enter credit for subject 2: 4
Enter marks for subject 3: 89
Enter credit for subject 3: 4
Enter marks for subject 4: 91
Enter credit for subject 4: 3
Enter marks for subject 5: 67
Enter credit for subject 5: 3
Enter marks for subject 6: 78
Enter credit for subject 6: 2
Enter marks for subject 7: 98
Enter credit for subject 7: 1
Enter marks for subject 8: 97
Enter credit for subject 8: 1
USN: 1BM23CS312
Name: Shashank
SGPA for sem 1: 9.045454545454545
CGPA: 9.045454545454545
```

LABORATORY PROGRAM – 3

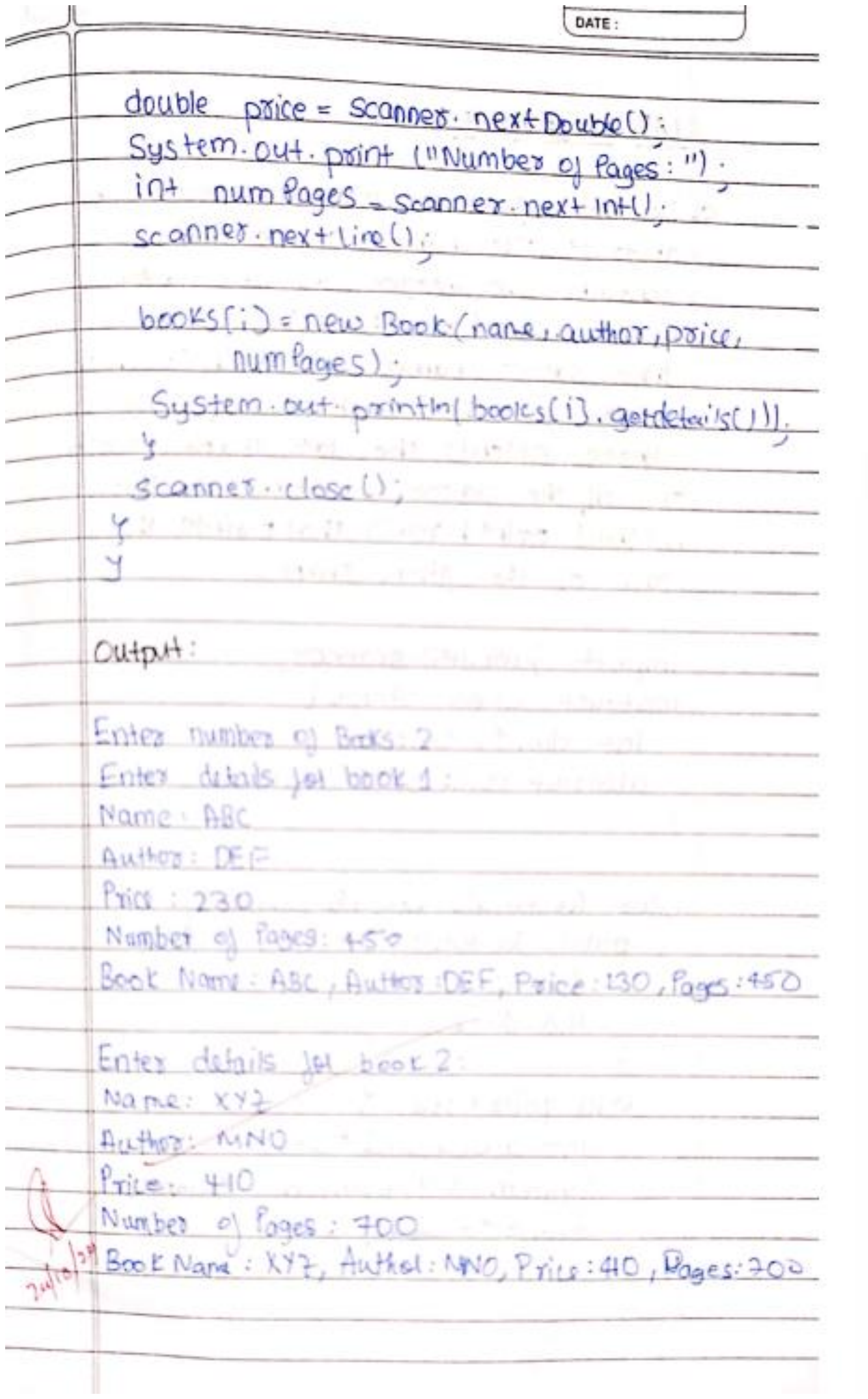
Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString( ) method that could display the complete details of the book. Develop a Java program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



Lab program 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{
    private string name;
    private string author;
    private double price;
    private int numPages;

    public Book(String name, String author, double price,
        int numPages){
            this.name = name;
            this.author=author;
            this.price= prince;
            this.numPages = numPages;
    }

    public void setDetails(String name, String author,
      double price, int numPages){
            this.name = name;
            this.author = author;
        this.price = price;
        this.numPages = numPages;
    }

    Public String getDetails() {
        return toString();
    }

    public String ~~tp ts~~ toString() {
        return "BookName:" + name + ", Author:" +
        author + ", Price: " + price + ", Pages:" +
        numPages;
    }
}

public class BookDemo {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in
        System.out.print("Enter no. of books: ");
        int n = scanner.nextInt();
        scanner.nextLine();

        Book[] books = new Book[n];

        for (int i=0; i<n; i++) {
            System.out.println("Ente " + (i+1) + ":
            System.out.print("Name: ");
            String name = scanner.nextLine();
            System.out.print("Author:");
            String author = scanner.nextLine();
            System.out.print("Price: ");
```



DATE :

```
double price = scanner.nextDouble();
System.out.print ("Number of Pages : ");
int numPages = scanner.nextInt();
scanner.nextLine();

books[i] = new Book(name, author, price,
        numPages);
 System.out.println(books[i].getdetails());
}
scanner.close();
}
}
```

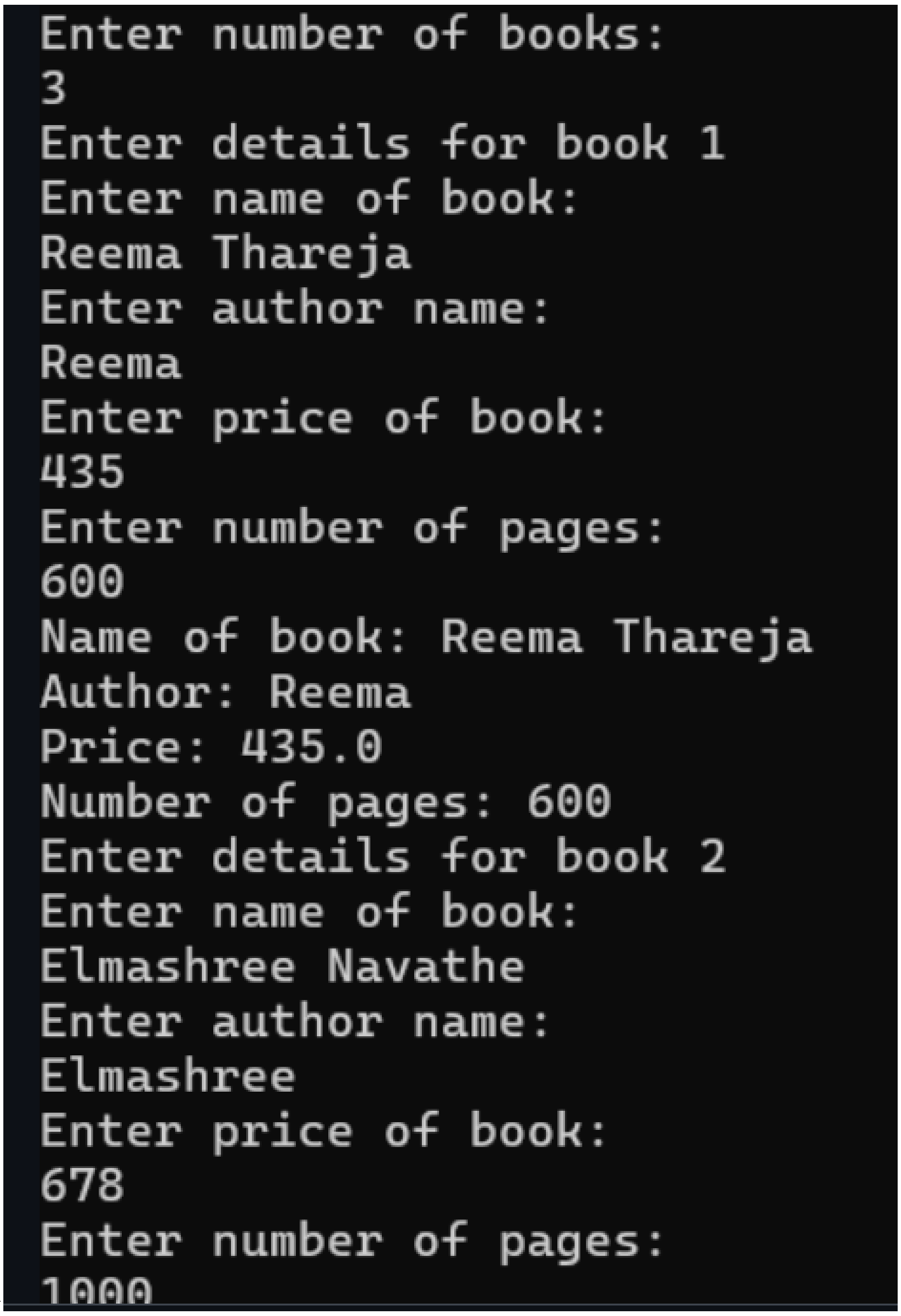
Output:

```
Enter number of Books: 2
Enter details for book 1:
Name: ABC
Author: DEF
Price : 230
Number of Pages: 450
Book Name: ABC, Author :DEF, Price: 130, Pages: 450

Enter details for book 2:
Name: XYZ
Author: MNO
Price: 410
Number of Pages : 700
Book Name : XYZ, Authol: MNO, Price: 410, Pages: 700
```

24/10/24

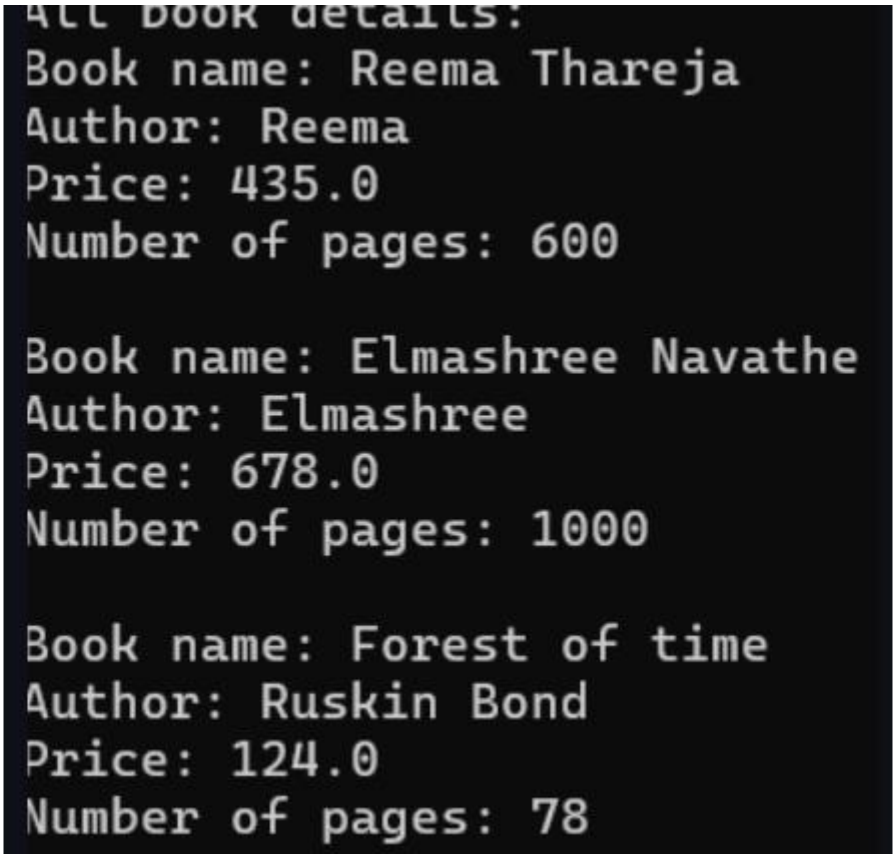
import

```
java.util.Scanner;
class Book {
    String name, author;
    double price;
    int noPage;
    Book() {}
    Book(String name, String author, double price, int noPage) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.noPage = noPage;
    }
    void setDetails() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter name of book: ");
        name = sc.nextLine();
        System.out.println("Enter author name: ");
        author = sc.nextLine();
        System.out.println("Enter price of book: ");
        price = sc.nextDouble();
        System.out.println("Enter number of pages: ");
        noPage = sc.nextInt();
    }
    void getDetails() {
        System.out.println("Name of book: " + name);
        System.out.println("Author: " + author);
        System.out.println("Price: " + price);
        System.out.println("Number of pages: " + noPage);
    }
    public String toString() {
        return "Book name: " + name + "\n" + "Author: " + author + "\n" + "Price: " +
price + "\n" + "Number of pages: " + noPage + "\n";
    }
}
class MyBook {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of books: ");
        int n = sc.nextInt();
        sc.nextLine();
        Book[] books = new Book[n];
        for (int i = 0; i < n; i++) {
            books[i] = new Book();
            System.out.println("Enter details for book " + (i + 1));
            books[i].setDetails();
            books[i].getDetails();
        }
        System.out.println("All book details: ");
        for (Book book : books) {
            System.out.println(book);
        }
    }
```



```
Enter number of books:
3
Enter details for book 1
Enter name of book:
Reema Thareja
Enter author name:
Reema
Enter price of book:
435
Enter number of pages:
600
Name of book: Reema Thareja
Author: Reema
Price: 435.0
Number of pages: 600
Enter details for book 2
Enter name of book:
Elmashree Navathe
Enter author name:
Elmashree
Enter price of book:
678
Enter number of pages:
1000
```

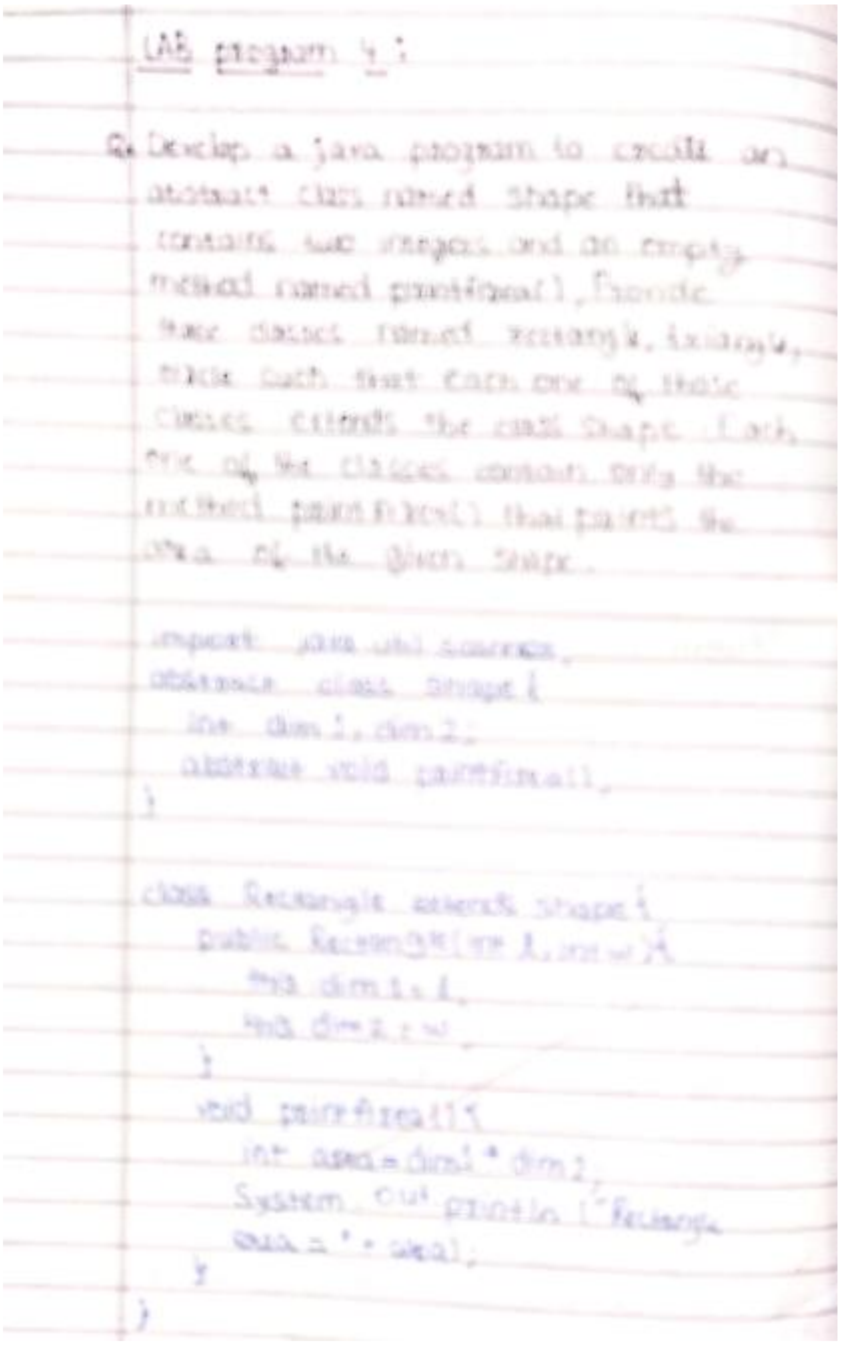
}



```
All book details:
Book name: Reema Thareja
Author: Reema
Price: 435.0
Number of pages: 600

Book name: Elmashree Navathe
Author: Elmashree
Price: 678.0
Number of pages: 1000

Book name: Forest of time
Author: Ruskin Bond
Price: 124.0
Number of pages: 78
```

LABORATORY PROGRAM – 4

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

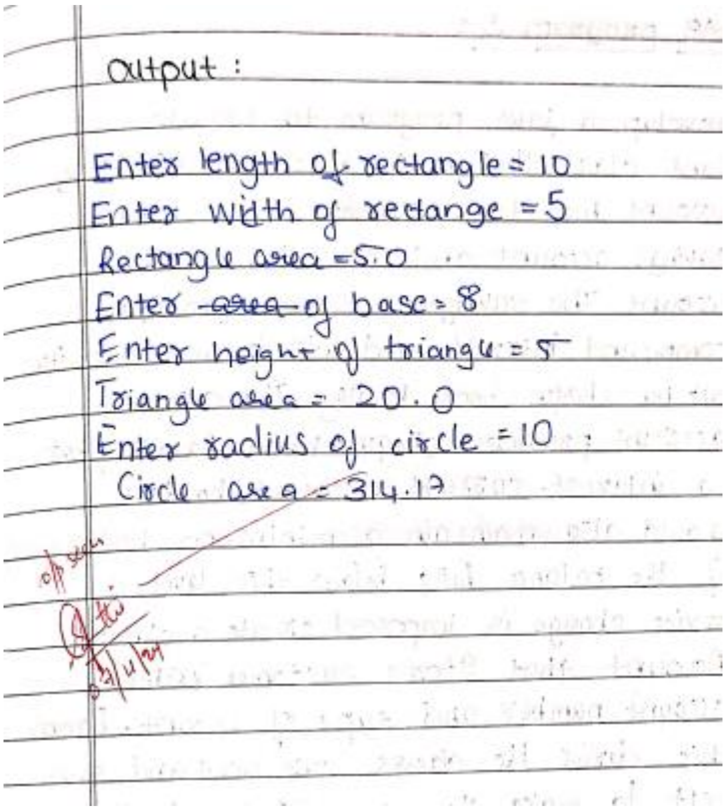


LAB program 4 :

Q. 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, Circle such that each one of these 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;
abstract class Shape {
  int dim1, dim2;
  abstract void printArea();
}

class Rectangle extends Shape {
    public Rectangle(int l, int w){
        this.dim1 = l;
        this.dim2 = w;
    }
    void printArea(){
        int area = dim1 * dim2;
        System.out.println("Rectangle
        area = " + area);
    }
}
```

```
class Triangle extends shape{
    public Triangle (int b, int h){
        this.dim 1 = b;
        this.dim 2 = h;
    }
    void printArea(){
        float area = 0.5 * dim 1 * dim2;
        System.out.println("Triangle area= "+
        area);
    }
}

class circle extends shape{
    ~~private fin~~
    public circle (int r){
        this.dim 1 = r;
    }
    void print Area(){
        float area = Math.pi * dim1 * dim1;
        System.out.println( "circle area:"
        + area);
    }
}
```

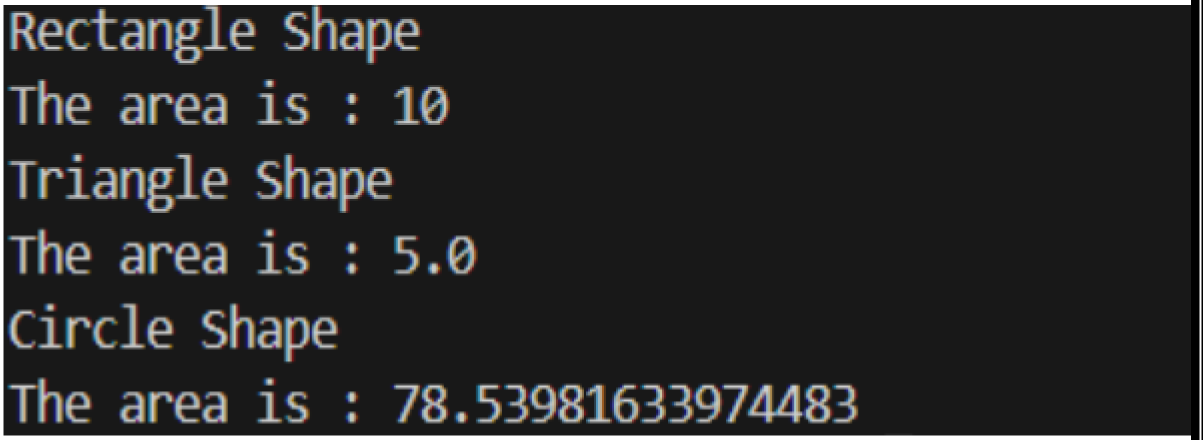


```
output :

Enter length of rectangle = 10
Enter width of rectange = 5
Rectangle area = 5.0
Enter ~~area~~ of base = 8
Enter height of triangle = 5
Triangle area = 20.0
Enter radius of circle = 10
 Circle area = 314.17
```

o/p seen

[illegible]

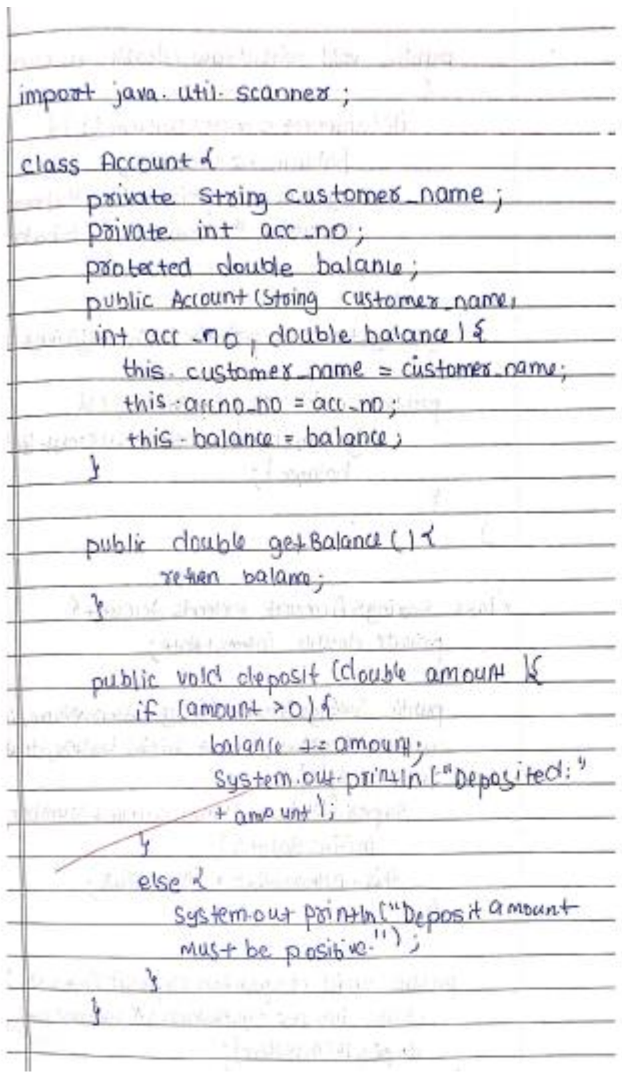
```
import java.util.Scanner;


abstract class Shape {
    int dimension1;
    int dimension2;



    abstract void printArea();
}

class Rectangle extends Shape {



    public Rectangle(int length, int width) {
        this.dimension1 = length;
        this.dimension2 = width;
    }
    void printArea() {
        int area = dimension1 * dimension2;
        System.out.println("Rectangle Area: " + area);
    }
}

class Triangle extends Shape {


    public Triangle(int base, int height) {
        this.dimension1 = base;
        this.dimension2 = height;
    }



    void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        System.out.println("Triangle Area: " + area);
    }
}

class Circle extends Shape {
    private final double pi = 3.14159;


    public Circle(int radius) {
        this.dimension1 = radius;
        this.dimension2 = 0;
    }
    void printArea() {
        double area = pi * dimension1 * dimension1;
        System.out.println("Circle Area: " + area);
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);



        System.out.print("Enter length of rectangle: ");
        int length = scanner.nextInt();
        System.out.print("Enter width of rectangle: ");
        int width = scanner.nextInt();
        Rectangle rectangle = new Rectangle(length,
width);
        rectangle.printArea();


        System.out.print("Enter base of triangle: ");
        int base = scanner.nextInt();
        System.out.print("Enter height of triangle: ");
        int height = scanner.nextInt();
        Triangle triangle = new Triangle(base, height);
        triangle.printArea();

        System.out.print("Enter radius of circle: ");
        int radius = scanner.nextInt();
        Circle circle = new Circle(radius);
        circle.printArea();

        scanner.close();
    }
}
```



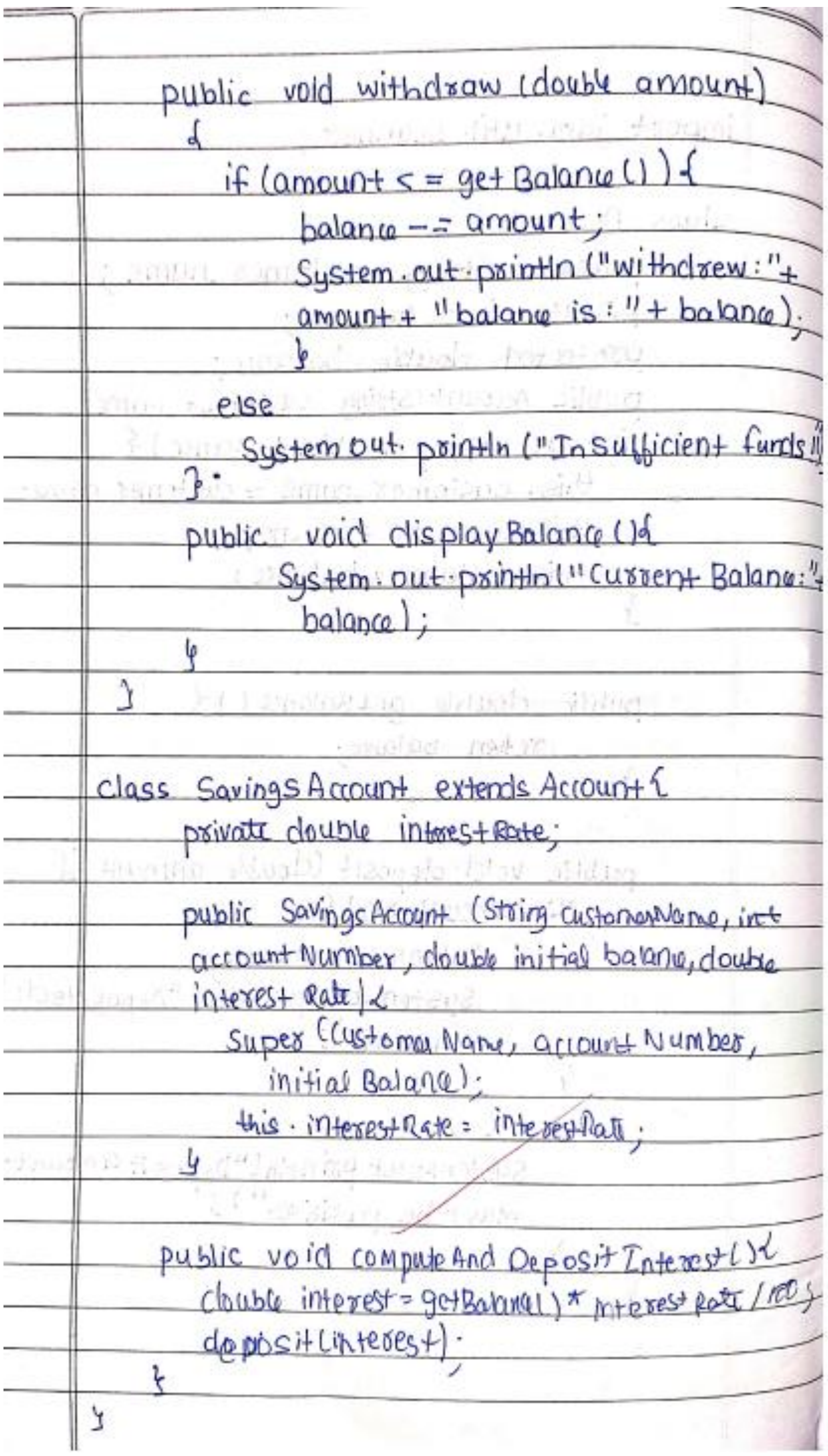
```
Rectangle Shape
The area is : 10
Triangle Shape
The area is : 5.0
Circle Shape
The area is : 78.53981633974483
```

LABORATORY PROGRAM – 5

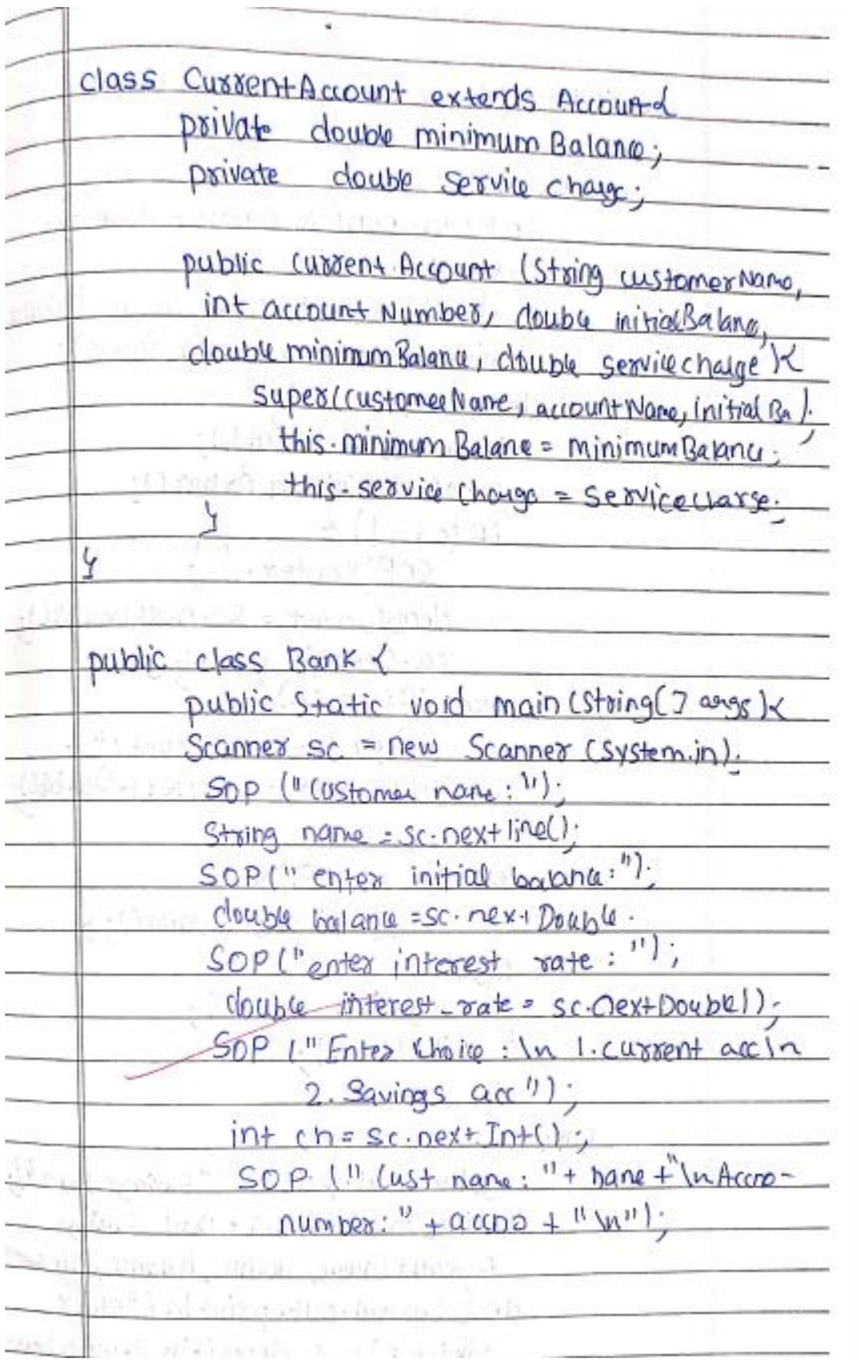
Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book 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.



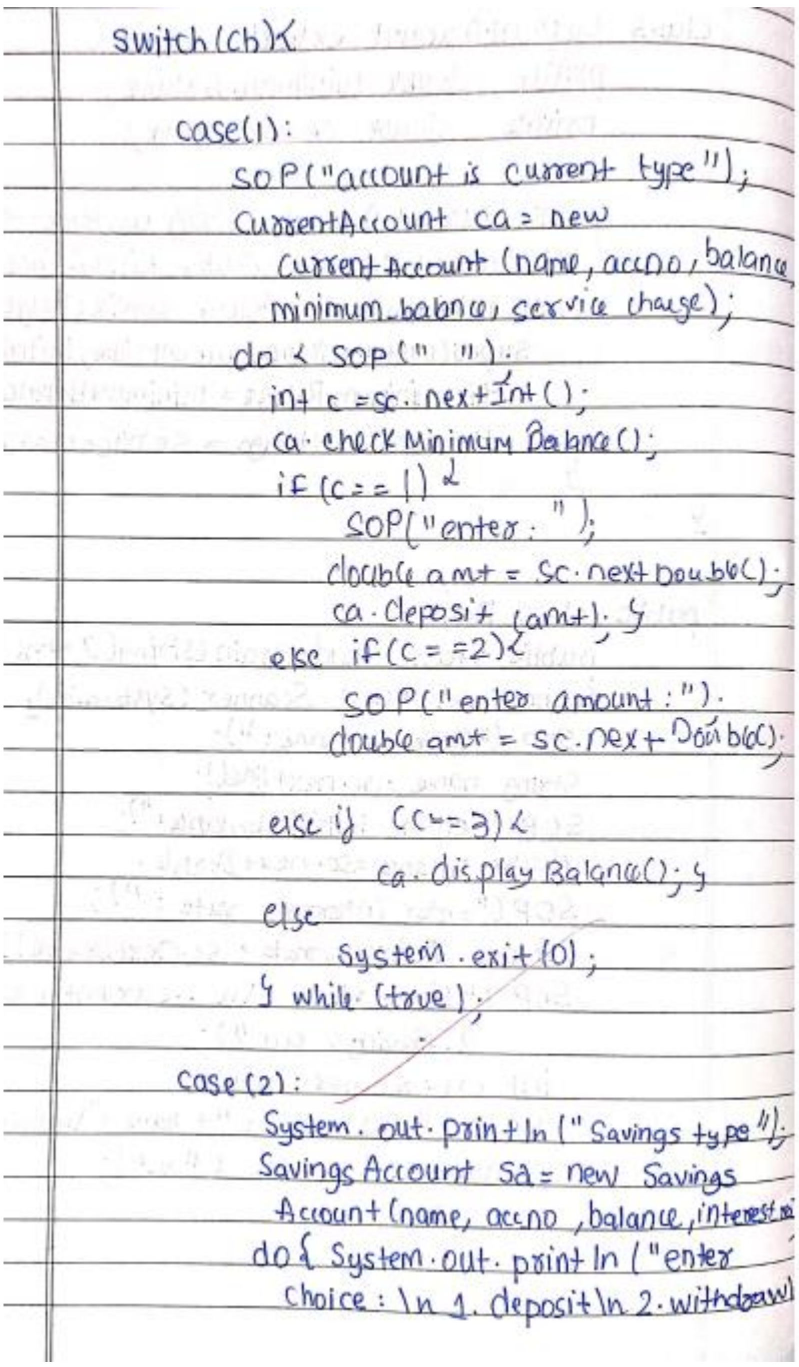
```
import java.util.scanner;

class Account {
    private String customer_name;
    private int acc_no;
    protected double balance;
    public Account(String customer_name,
     int acc_no, double balance) {
       this.customer_name = customer_name;
       this.accno_no = acc_no;
       this.balance = balance;
    }

    public double getBalance() {
          return balance;
    }

    public void deposit (double amount) {
        if (amount >0) {
              balance += amount;
              System.out.println("Deposited:"
               + amount);
        }
        else {
            system.out.println("Deposit amount
            must be positive.");
        }
    }
```



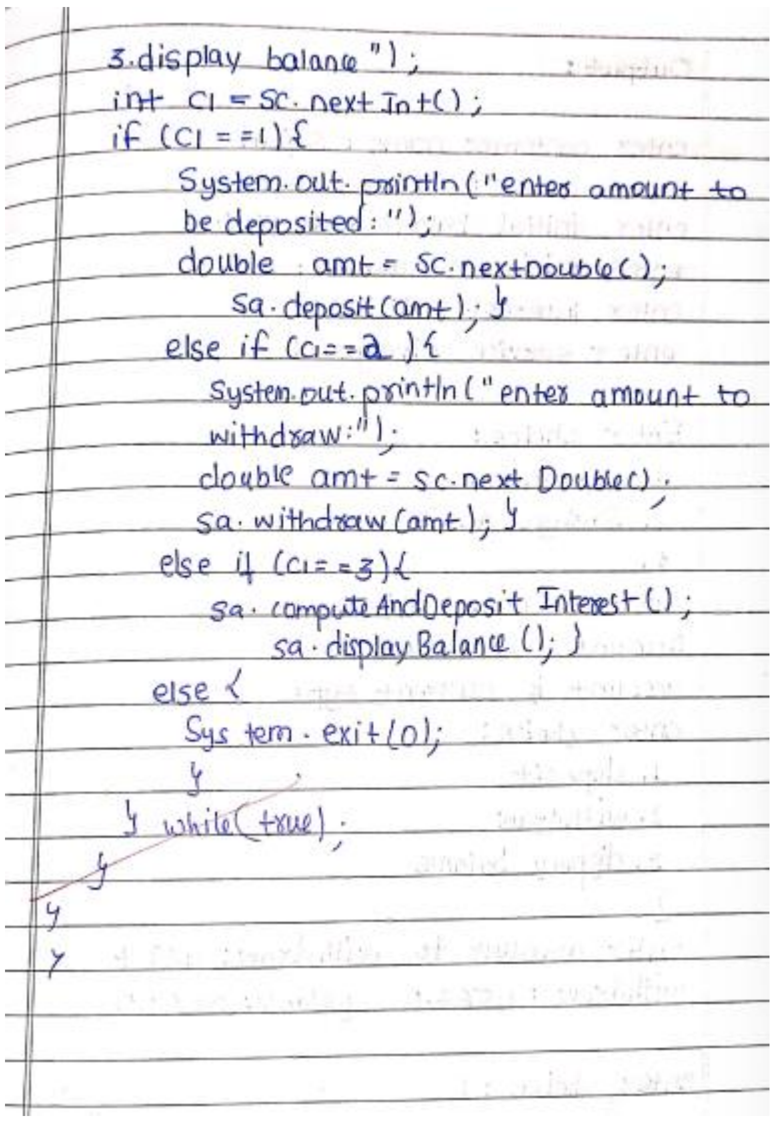
```
    public void withdraw (double amount)
    {
        if (amount <= getBalance()) {
            balance -= amount;
            System.out.println ("withdrew:"+
            amount + "balance is:" + balance);
        }
        else
          System.out.println ("Insufficient funds!");
    }.
    public void displayBalance(){
        System.out.println("Current Balance:"+
          balance);
    }
}
class SavingsAccount extends Account{
    private double interestRate;

    public SavingsAccount (String customerName, int
    accountNumber, double initial balance, double
    interest Rate){
        Super (CustomerName, accountNumber,
          initial Balance);
        this.interestRate = interestRate;
    }

    public void computeAndDepositInterest(){
      double interest = getBalance() * interestRate / 100;
      deposit(interest);
    }
}
```



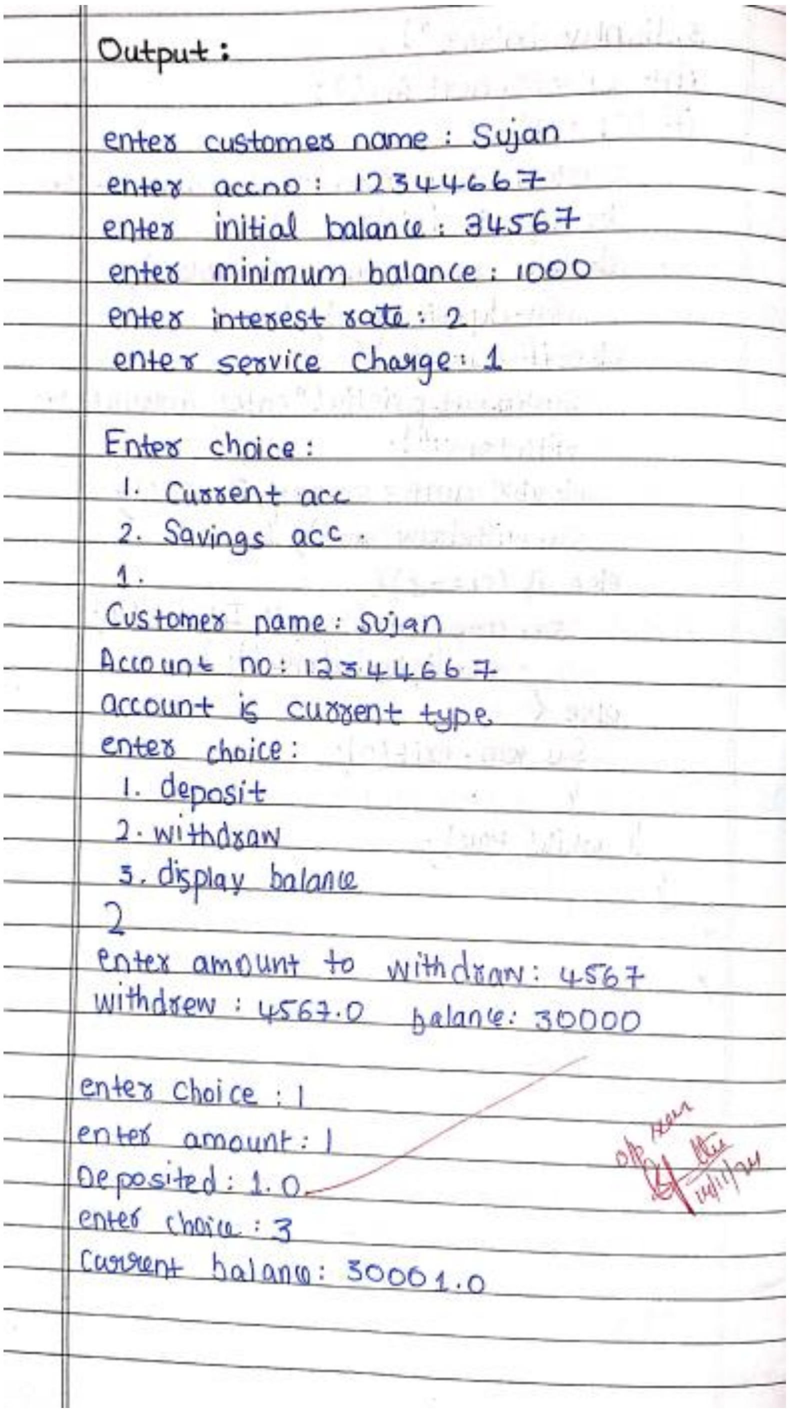
```
class CurrentAccount extends Account{
      private double minimumBalance;
      private double Service charge;

      public current Account (String customerName,
        int account Number, double initialBalance,
        double minimumBalance; double servicecharge ){
            super(customerName, accountName, initial Ba ).
            this.minimum Balane = minimumBalance;
            this. service charge = Servicecharse.
      }
}

public class Bank {
      public static void main (String[] args){
      Scanner sc = new Scanner (System.in).
       SOP ("Customer name: ");
       String name = sc.nextline();
       SOP(" enter initial balance: ");
       double balance =sc. next Double.
       SOP("enter interest rate : ");
        double interest_rate = sc.nextDouble());
        SOP ("Enter Choice : \n 1.current acc\n
              2. Savings acc ");
         int ch= sc.next Int();
         SOP ("Cust name: "+ name + "\nAccno-
            number: " +accno + "\n");
```



```
switch(ch){

    case(1):
        SOP("account is current type");
        CurrentAccount ca = new
          Current Account (name, accno, balance
          minimum_balance, service charge);
        do { SOP(" ");
          int c=sc.nextInt();
          ca.checkMinimum Balance();
          if(c==1){
              SOP("enter: ");
              double amt = sc.nextDouble();
              ca.deposit(amt); }
          else if(c==2){
              SOP("enter amount:");
              double amt = sc.next Double();

          else if (c==3){
                 ca.display Balance(); }
          else
              System.exit(0);
          } while (true);

    case(2):
        System.out.println("Savings type");
        Savings Account sa = new Savings
          Account(name, accno, balance, interest
        do{ System.out.println("enter
          choice : \n 1. deposit\n 2. withdraw
```



```
3.display balance");
int c1 = sc.nextInt();
if (c1 ==1){
    System.out.println("enter amount to
    be deposited:");
    double amt = sc.nextDouble();
        sa.deposit(amt); }
  else if (c1==2){
     System.out.println("enter amount to
     withdraw:");
     double amt = sc.next Double();
     sa.withdraw(amt); }
  else if (c1==3){
     sa.computeAndDeposit Interest();
          sa.displayBalance(); }
  else {
    System.exit(0);
     }
 } while(true);
  }
}
}
```

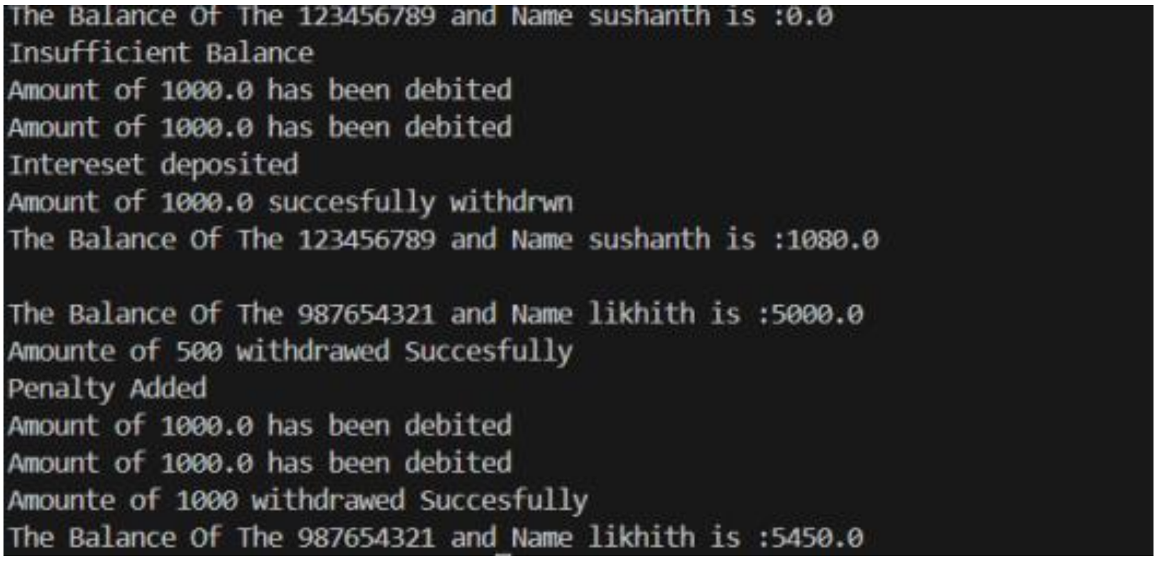


```
Output:

enter customer name : Sujan
enter accno : 12344667
enter initial balance : 34567
enter minimum balance : 1000
enter interest rate : 2
enter service charge : 1
Enter choice :
 1. Current acc
 2. Savings acc .
 1.
Customer name : Sujan
Account no : 12344667
account is current type.
enter choice :
 1. deposit
 2. withdraw
 3. display balance
2
enter amount to withdraw : 4567
withdrew : 4567.0   balance : 30000

enter choice : 1
enter amount : 1
Deposited : 1.0
enter choice : 3
Current balance : 30001.0
```

[illegible]

```
import java.util.Scanner;

class Account {
    private String customer_name;
    private int acc_no;
    protected double balance;
    public Account(String customer_name, int acc_no,
double balance) {
        this.customer_name = customer_name;
        this.acc_no = acc_no;
        this.balance = balance;
    }

    public double getBalance() {
        return balance;
    }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited: " + amount);
        } else {
            System.out.println("Deposit amount must be
positive.");
        }
    }
   public void withdraw(double amount)
     {
       if(amount<=getBalance()){
          balance-=amount;
          System.out.println("withdrew:"+amount + "
balance is:"+ balance);
          }
       else
         System.out.println("Insufficient funds!!");
     }
    public void displayBalance(){
        System.out.println("Current Balance: " +
balance);
```

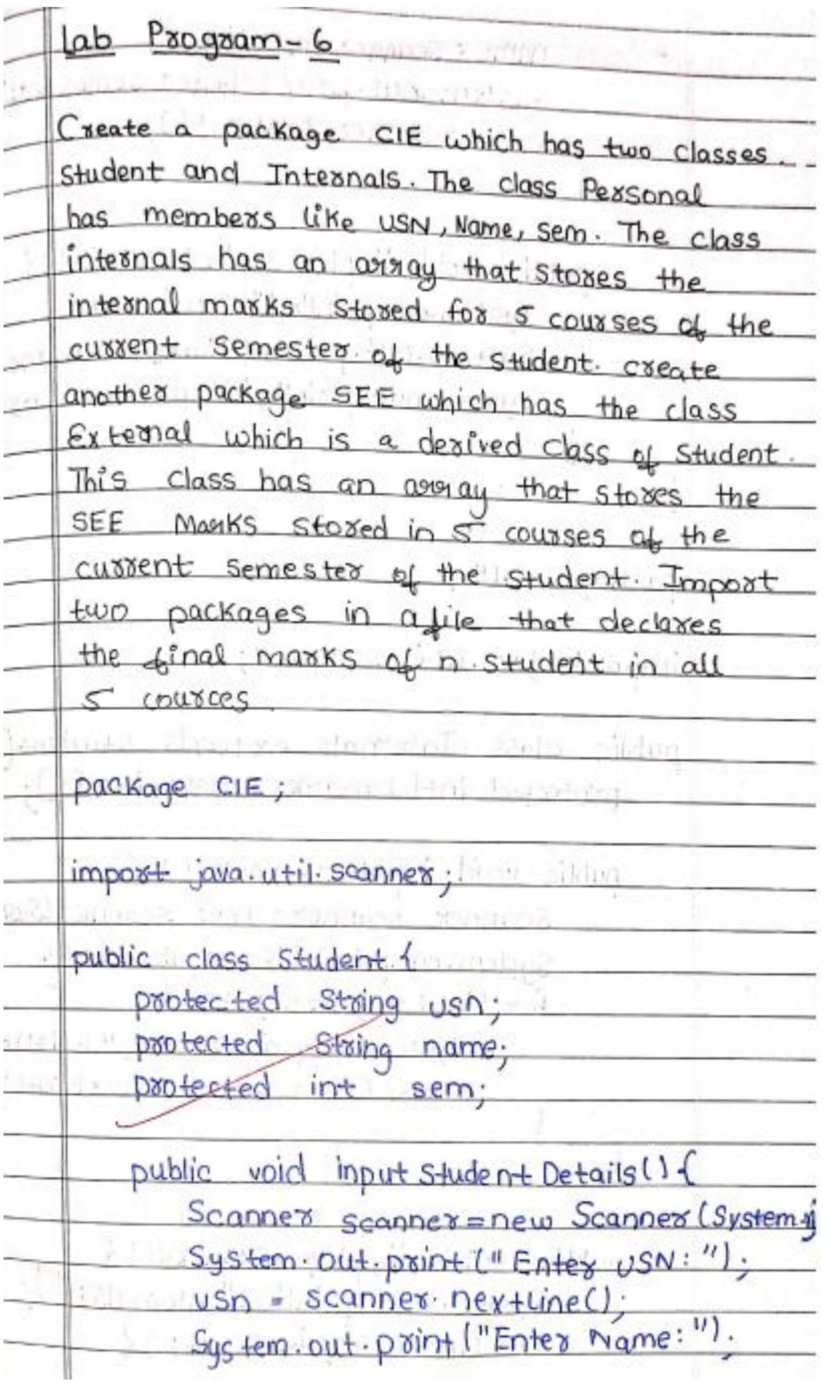
```
    }
}

class SavingsAccount extends Account {
    private double interestRate;

    public SavingsAccount(String customerName, int
accountNumber, double initialBalance, double
interestRate) {
        super(customerName, accountNumber,
initialBalance);
        this.interestRate = interestRate;
    }

    public void computeAndDepositInterest() {
        double interest = getBalance() * interestRate /
100;
        deposit(interest);
    }
}
class CurrentAccount extends Account {
    private double minimumBalance;
    private double serviceCharge;

    public CurrentAccount(String customerName, int
accountNumber, double initialBalance, double
minimumBalance, double serviceCharge) {
        super(customerName, accountNumber,
initialBalance);
        this.minimumBalance = minimumBalance;
        this.serviceCharge = serviceCharge;
    }
    public void checkMinimumBalance() {
        if (getBalance() < minimumBalance) {
          System.out.println("Balance is below
minimum");
          balance-=serviceCharge;
          System.out.println("Deducted service charge:"
+serviceCharge);
          System.out.println("Balance after deduction
is:"+balance);
       }
    }
}

public class Bank {
    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 accno:");
        int acc_no=sc.nextInt();
        System.out.println("enter initial balance:");
        double balance=sc.nextDouble();
        System.out.println("enter minimum balance:");
        double minimum_balance=sc.nextDouble();
        System.out.println("enter interest rate:");
        double interest_rate=sc.nextDouble();
        System.out.println("enter service charge:");
        double service_charge=sc.nextDouble();
        System.out.println("Enter choice:\n 1.Current
acc\n 2.Savings acc");
        int ch=sc.nextInt();
        System.out.println("Customer name is:"+
name+"\nAccount number:"+acc_no+"\n");

        switch(ch){
           case(1):
                System.out.println("account is current
type");
                CurrentAccount ca = new
CurrentAccount(name,acc_no,balance,minimum_bala
nce,service_charge);
               do{ System.out.println("enter choice:\n
1.deposit\n 2.withdraw\n 3.display balance");
                int c=sc.nextInt();
                ca.checkMinimumBalance();
                if(c==1){
                   System.out.println("enter amount to be
deposited:");
                   double amt=sc.nextDouble();
                     ca.deposit(amt);}
                else if(c==2){
                   System.out.println("enter amount to
withdraw:");
                   double amt=sc.nextDouble();
                   ca.withdraw(amt);}
                else if(c==3){
                   ca.displayBalance();}
                else
                  System.exit(0);
                 }while(true);

           case(2):
                 System.out.println("account is savings
type");
                SavingsAccount sa=new
SavingsAccount(name,acc_no,balance,interest_rate);
                do{ System.out.println("enter choice:\n
1.deposit\n 2.withdraw\n 3.display balance");
                int c1=sc.nextInt();
                if(c1==1){
                   System.out.println("enter amount to be
deposited:");
                   double amt=sc.nextDouble();
                     sa.deposit(amt);}
                else if(c1==2){
                   System.out.println("enter amount to
withdraw:");
                   double amt=sc.nextDouble();
                   sa.withdraw(amt);}
                else if(c1==3){
                 sa.computeAndDepositInterest();
                   sa.displayBalance();}
                else{
                  System.exit(0);
                      }
                }while(true);
   }
}
}
```



```
The Balance Of The 123456789 and Name sushanth is :0.0
Insufficient Balance
Amount of 1000.0 has been debited
Amount of 1000.0 has been debited
Intereset deposited
Amount of 1000.0 succesfully withdrwn
The Balance Of The 123456789 and Name sushanth is :1080.0

The Balance Of The 987654321 and Name likhith is :5000.0
Amounte of 500 withdrawed Succesfully
Penalty Added
Amount of 1000.0 has been debited
Amount of 1000.0 has been debited
Amounte of 1000 withdrawed Succesfully
The Balance Of The 987654321 and Name likhith is :5450.0
```

LABORATORY PROGRAM – 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.



lab Program-6

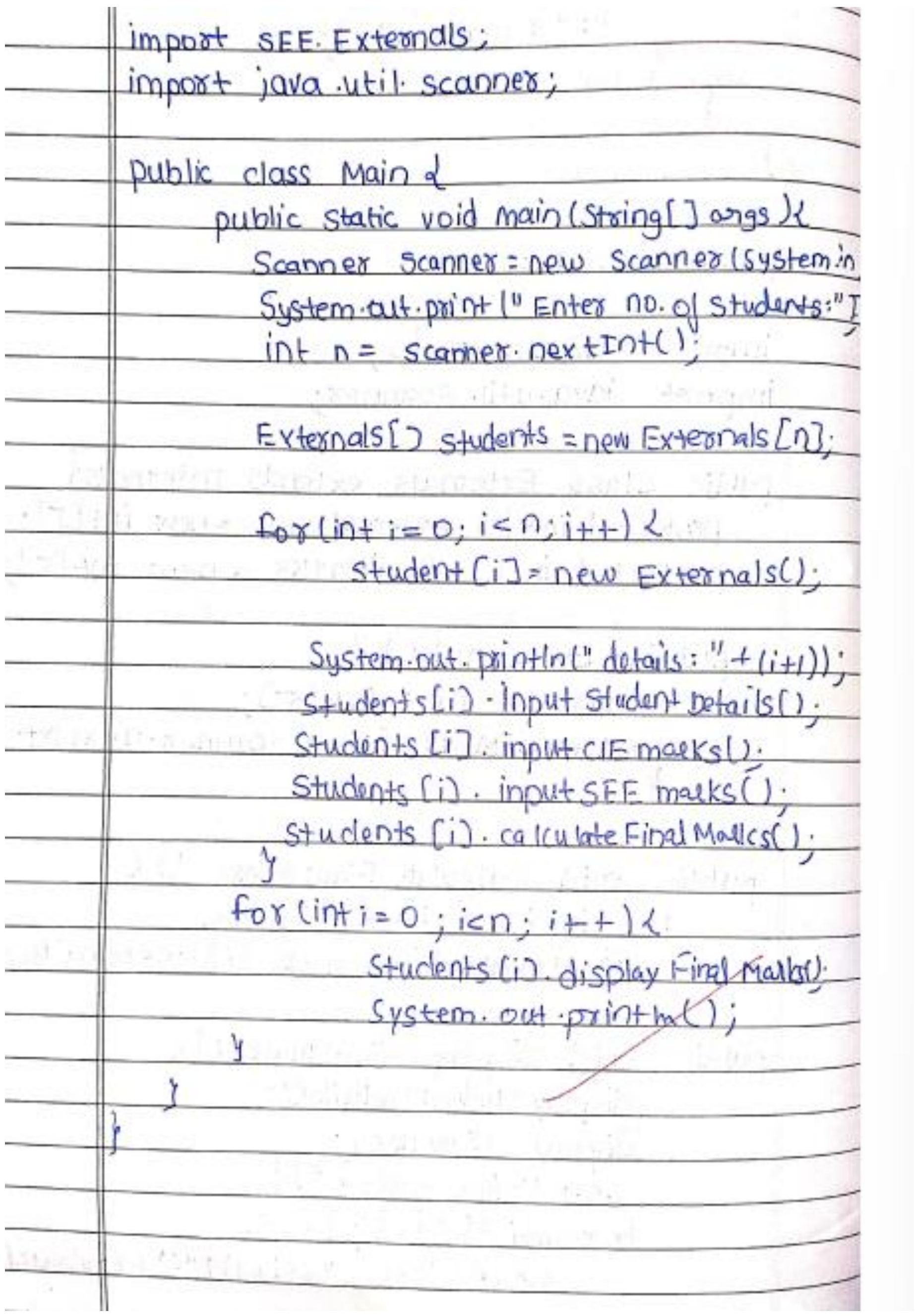
Create a package CIE which has two classes. Student and Internals. The class Personal has members like USN, Name, sem. The class internals has an array that stores the internal marks stored for 5 courses of the current Semester of the student. create another package SEE which has the class External which is a derived class of Student. This class has an array that stores the SEE Marks stored in 5 courses of the current semester of the student. Import two packages in a file that declares the final marks of n student in all 5 cources.

```
package CIE;

import java.util.scanner;

public class Student {
    protected String usn;
    protected String name;
    protected int sem;

    public void inputStudentDetails() {
        Scanner scanner=new Scanner(System.i
        System.out.print("Enter USN: ");
        usn = scanner.nextLine();
        System.out.print("Enter Name: ");
```

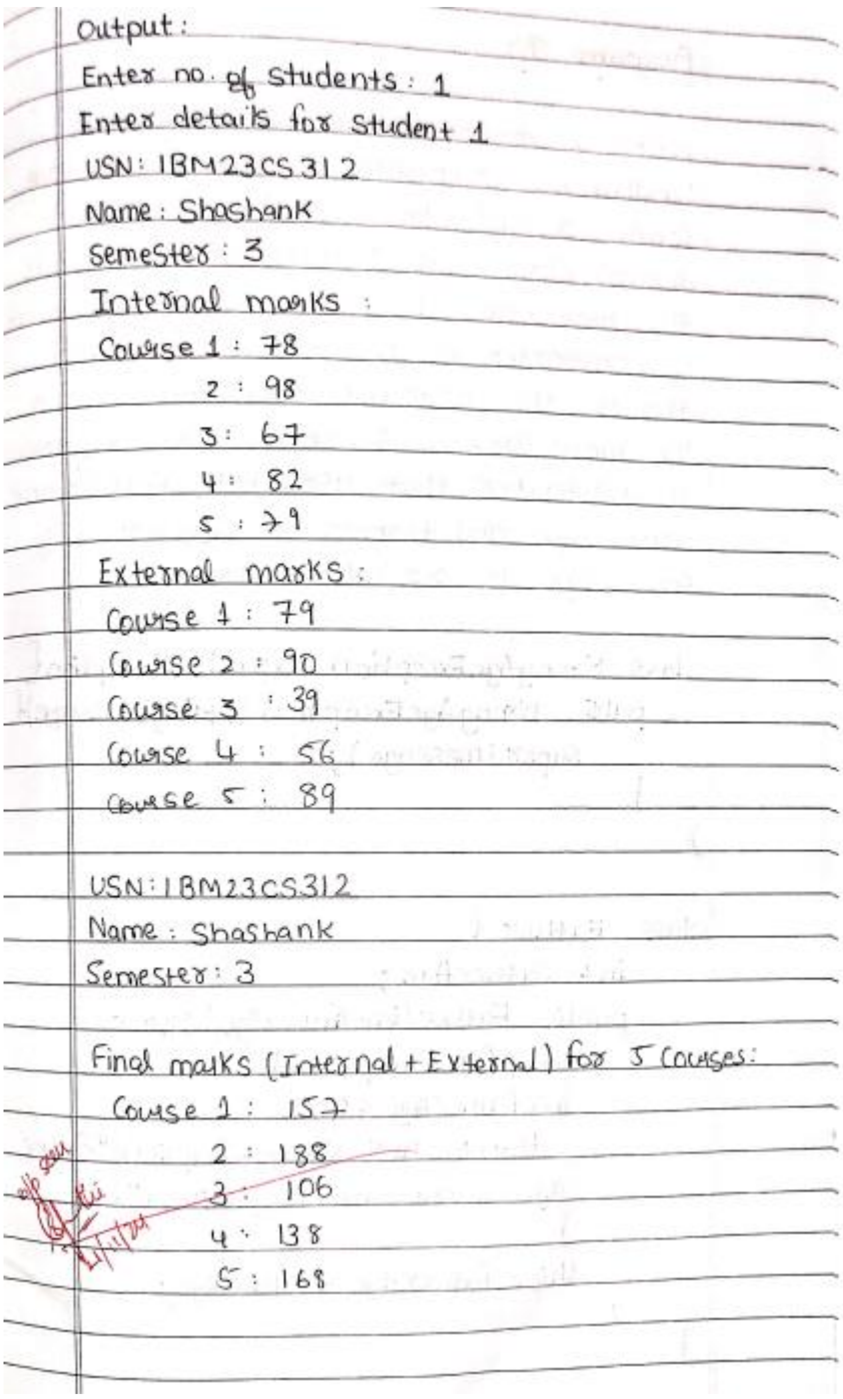
DATE :

```
        name = scanner.nextline();
        System.out.print("Enter Semester:")
        Sem = Scanner.nextInt();
    }

    public void displayStudentDetails(){
        System.out.println("USN: " + usn);
        System.out.println("Name : " + name);
        System.out.println("Semester:" + sem
    }
}

package CIE;

import java.util.scanner;

public class Internals extends student{
    protected int[] marks = new int [5];

    public void inputCIEmarks() {
        Scanner scanner = new scanner(System
        System.out.println("Enter internal: ").
        for (int i=0; i<5; i++){
            System.out.print ("Marks" + (i+1)+"
            marks [i] = scanner.nextInt();
        }
    }

    public void displayCIEmarks() {
        System.out.println("Internals: ");
        for(int i = 0; i< 5; i++){
```

```
        System.out.println("Course" + (i+1)+
        ":" + marks[i]);
      }
   }
}

package SEE;

import CIE.Internals;
import java.util.scanner;

public class Externals extends Internals{
   protected int[] externalMarks = new int[5];
   protected int[] finalMarks = new int[5];

   public Externals() {
       marks = new int[5];
       externalMarks[i] = Scanner.nextInt
     }

  public void calculate FinalMarks () <
      for (int i=0; i<5; i++)<
         finalmarks [i] = marks [i]+externalMarks[i]

  public void display finalMarks()<
         display StudentDetails();
         display CIEmarks();
         S.O.P("Final marks: ");
         for(inti =0; i<5; i++)<
               S.O.P("Course"+(i+1)+":"+finalmarks[i])
   }
}
```



```
import SEE.Externals;
import java.util.scanner;

public class Main {
      public static void main(String[] args){
         Scanner scanner = new Scanner(system.in
         System.out.print(" Enter no. of students:");
         int n = scanner.nextInt();

         Externals[] students = new Externals[n];

         for(int i=0; i<n; i++) {
               student[i] = new Externals();

            System.out.println(" details : "+(i+1));
            students[i].Input Student Details();
            Students[i]. input CIE marks();
            Students[i]. input SEE marks();
            Students[i]. calculate Final Marks();
         }
         for(int i=0 ; i<n ; i++) {
               Students[i]. display Final Marks();
               System.out.println();
         }
      }
}
```



```
Output:
Enter no. of students: 1
Enter details for student 1
USN: 1BM23CS312
Name: Shashank
Semester: 3
Internal marks:
 Course 1: 78
        2: 98
        3: 67
        4: 82
        5: 79
External marks:
 Course 1: 79
 Course 2: 90
 Course 3: 39
 Course 4: 56
 Course 5: 89

USN: 1BM23CS312
Name: Shashank
Semester: 3

Final marks (Internal + External) for 5 courses:
 Course 1: 157
        2: 188
        3: 106
        4: 138
        5: 168
```

[illegible]

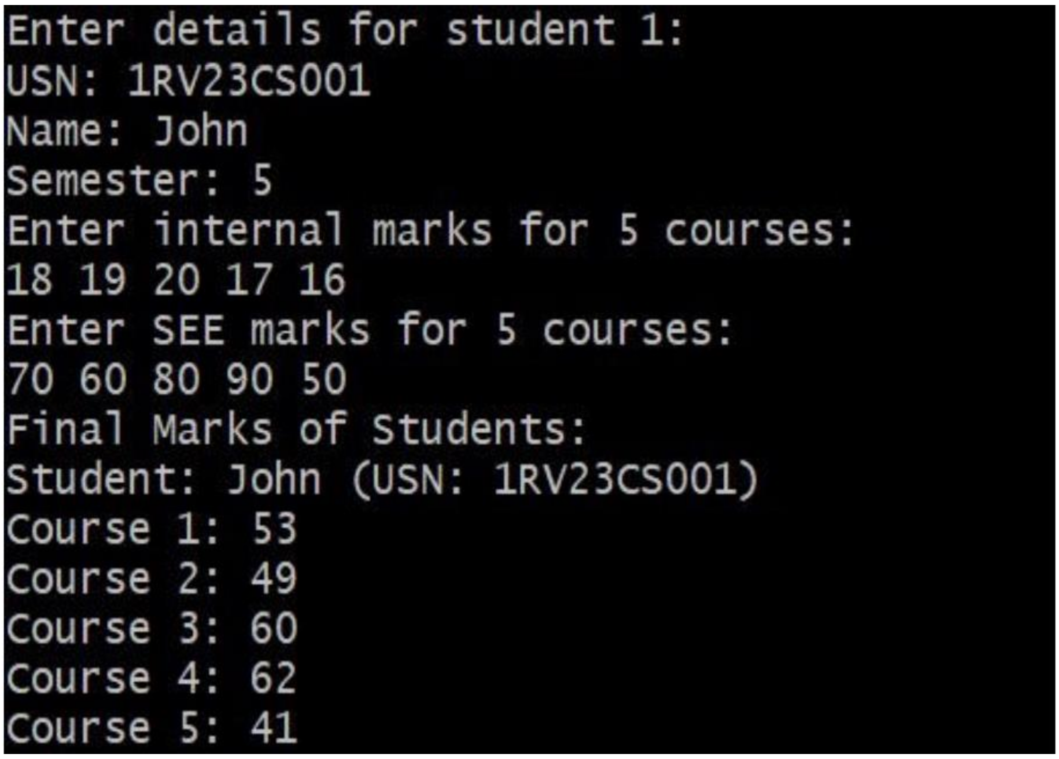
CIE/Student.java

```
package cie;
public class Student {
 public String usn;
 public String name;
 public int sem;
 public Student(String usn, String name, int sem) {
 this.usn = usn;
 this.name = name;
 this.sem = sem;
 }
}
File : CIE/Internal.java
package cie;
public class Internals {
 public int[] internalMarks = new int[5];
 public Internals(int[] marks) {
 if (marks.length == 5) {
 System.arraycopy(marks, 0, internalMarks, 0, 5);
 } else {
 System.out.println("Error: Please provide marks for
exactly 5
courses.");
 }
 }
}
```

```
File : SEE/External.java
package see;
import cie.Student;
public class External extends Student {
 public int[] externalMarks = new int[5];
 public External(String usn, String name, int sem, int[]
marks) {
 super(usn, name, sem);
 if (marks.length == 5) {
 System.arraycopy(marks, 0, externalMarks, 0, 5);
 } else {
 System.out.println("Error: Please provide marks for
exactly 5
courses.");
 }
 }
}
File : FinalMarrks.java
import cie.*;
import see.*;
import java.util.Scanner;
public class FinalMarks {
 public static void main(String[] args) {
 Scanner sc = new Scanner(System.in);
 System.out.print("Enter number of students: ");
 int n = sc.nextInt();
 Student[] students = new Student[n];
 Internals[] internals = new Internals[n];
```

```
 External[] externals = new External[n];
 for (int i = 0; i < n; i++) {
 System.out.println("Enter details for student " + (i +
1) + ":");
 System.out.print("USN: ");
 String usn = sc.next();
 System.out.print("Name: ");
 String name = sc.next();
 System.out.print("Semester: ");
 int sem = sc.nextInt();
 System.out.println("Enter internal marks for 5
courses:");
 int[] internalMarks = new int[5];
 for (int j = 0; j < 5; j++) {
 internalMarks[j] = sc.nextInt();
 }
 System.out.println("Enter SEE marks for 5 courses:");
 int[] externalMarks = new int[5];
 for (int j = 0; j < 5; j++) {
```

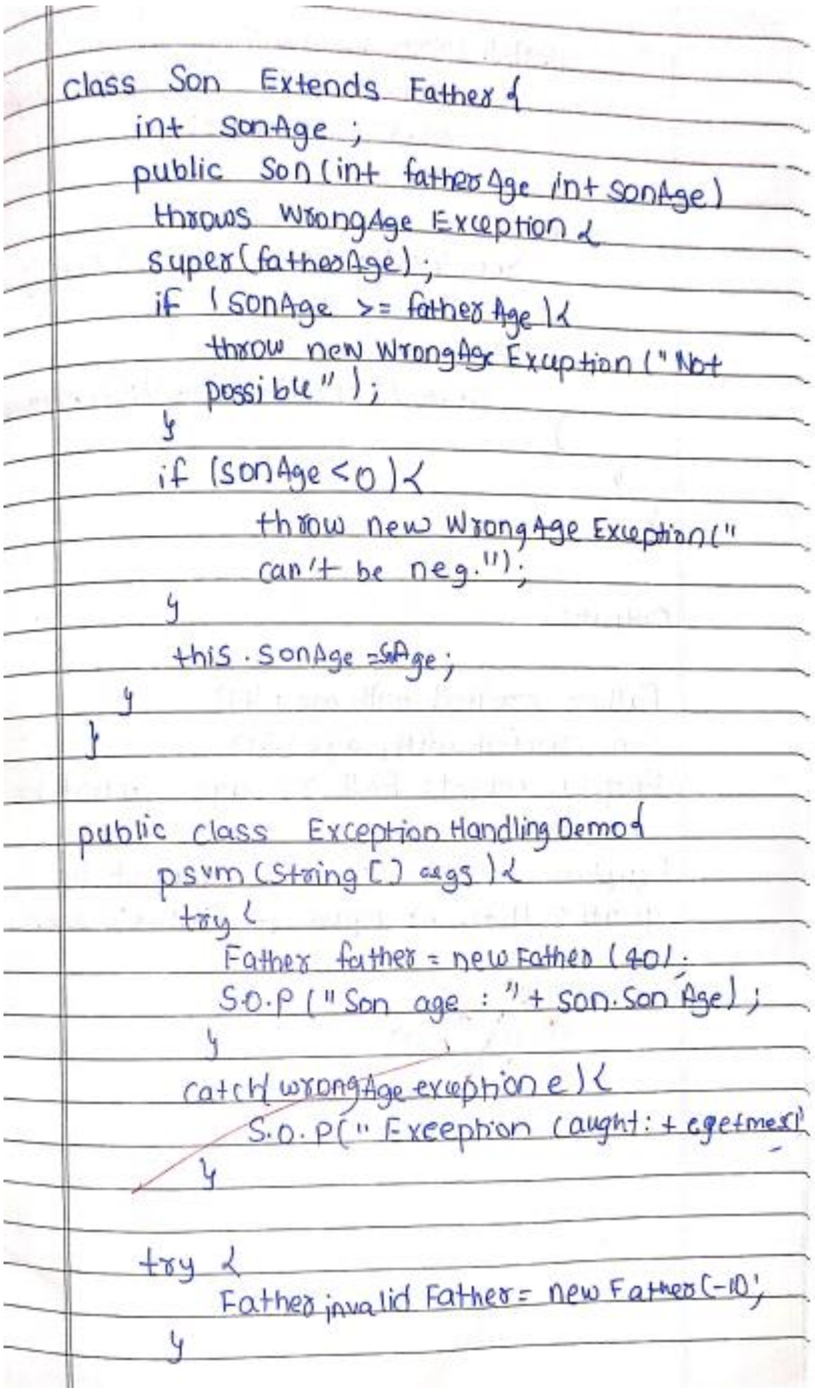
```
 externalMarks[j] = sc.nextInt();
 }
 students[i] = new Student(usn, name, sem);
 internals[i] = new Internals(internalMarks);
 externals[i] = new External(usn, name, sem,
externalMarks);
 }
 System.out.println("\nFinal Marks of Students:");
 for (int i = 0; i < n; i++) {
 System.out.println("Student: " + students[i].name + "
(USN: " +
students[i].usn + ")");
40
 for (int j = 0; j < 5; j++) {
 int finalMarks = internals[i].internalMarks[j] +
externals[i].externalMarks[j] / 2;
 System.out.println("Course " + (j + 1) + ": " +
finalMarks);
 }
 }
 sc.close();
 }
}
```



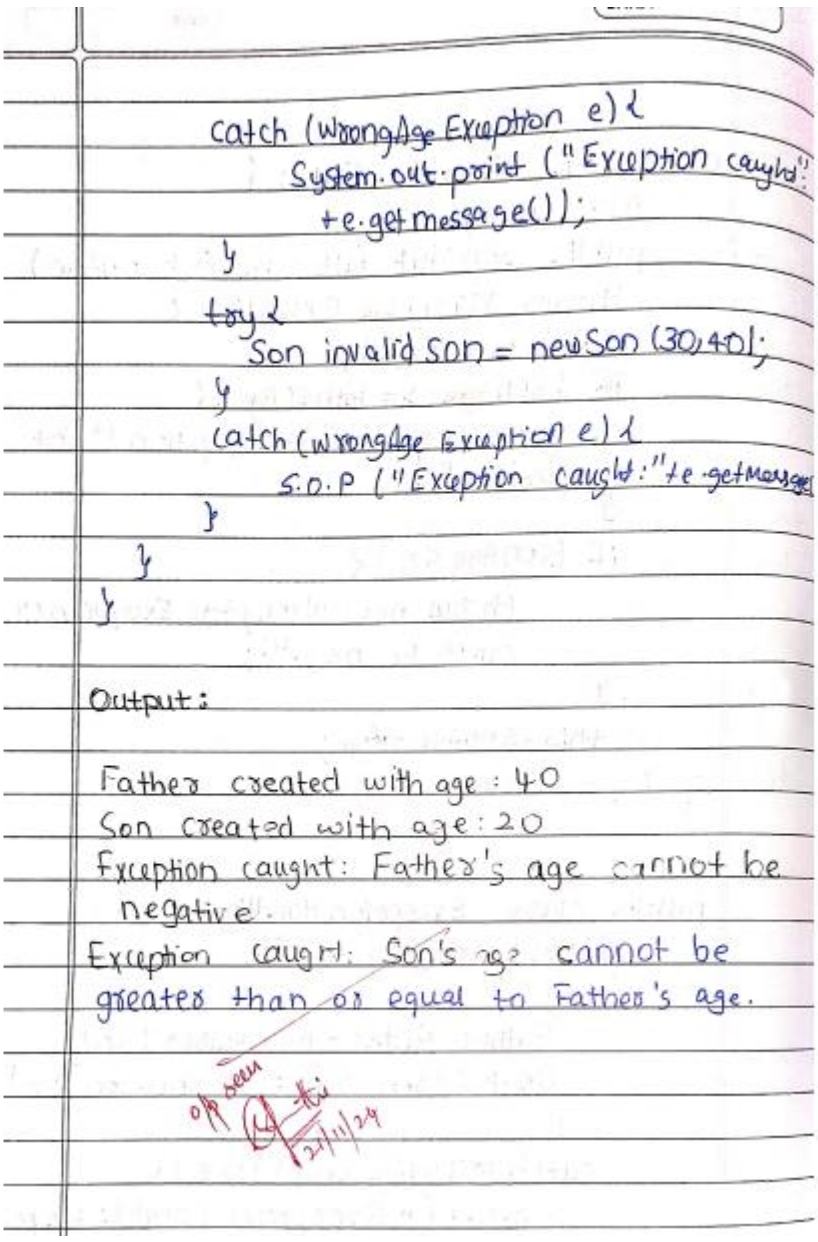
```
Enter details for student 1:
USN: 1RV23CS001
Name: John
Semester: 5
Enter internal marks for 5 courses:
18 19 20 17 16
Enter SEE marks for 5 courses:
70 60 80 90 50
Final Marks of Students:
Student: John (USN: 1RV23CS001)
Course 1: 53
Course 2: 49
Course 3: 60
Course 4: 62
Course 5: 41
```

LABORATORY PROGRAM – 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=father’s age.



```
class Son Extends Father {
    int sonAge ;
    public Son(int fatherAge int sonAge)
     throws WrongAge Exception {
     super(fatherAge);
     if (sonAge >= father Age){
         throw new WrongAge Exuption ("Not
         possible");
     }
     if (sonAge <0){
             throw new WrongAge Exception("
             can't be neg.");
     }
     this.sonAge =sAge;
  }
}

public class Exception Handling Demo{
     psvm(String[] args){
      try{
          Father father = new Father (40);
          S.O.P ("Son age : "+ son.Son Age);
          }
      catch(wrongAge exception e){
            S.O.P(" Exception caught: + e.getmess()
         }

    try {
          Father invalid Father= new Father(-10);
    }
```



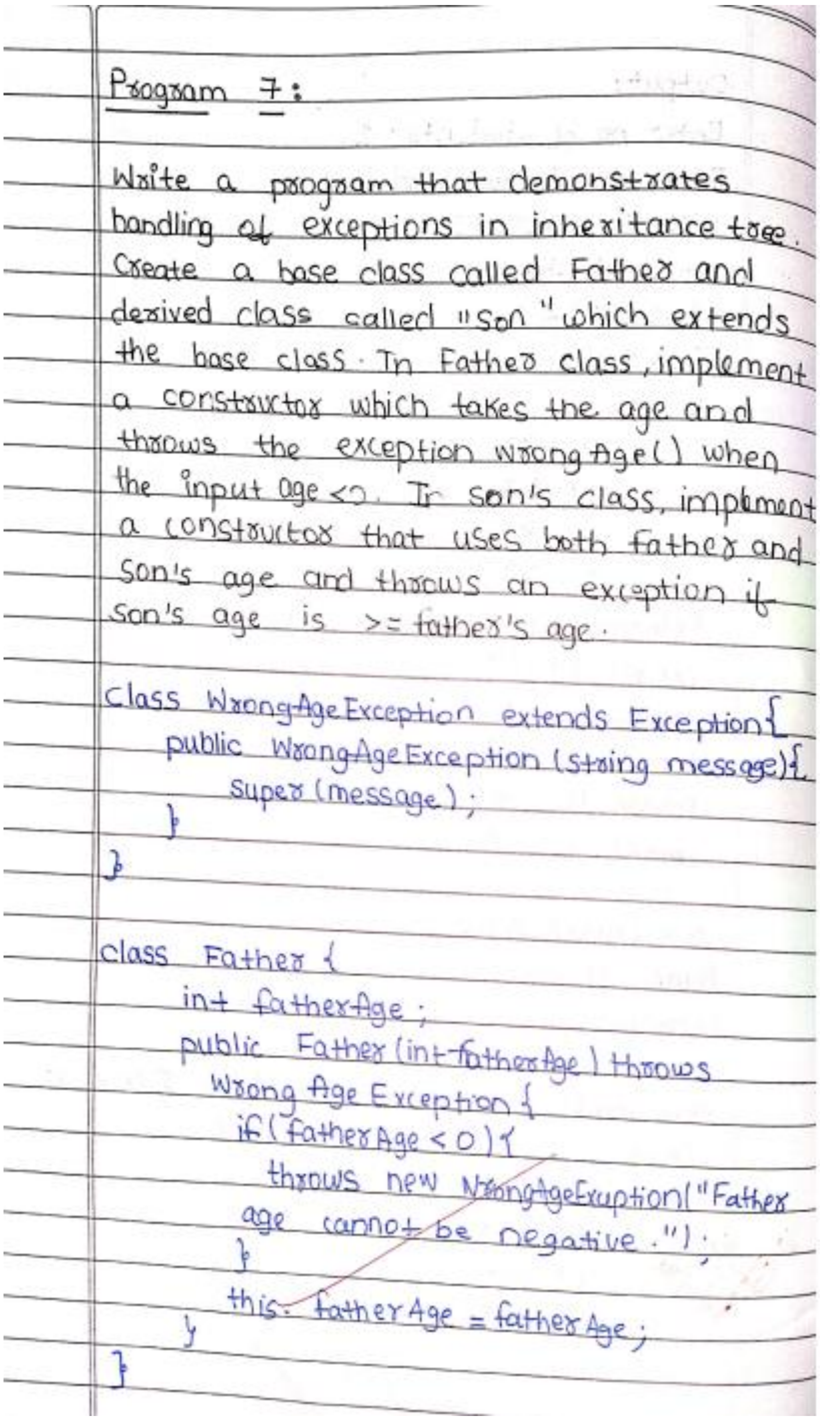
```
        catch (WrongAge Exception e){
            System.out.print ("Exception caught:"
                +e.getmessage());
        }
        try {
            Son invalid Son = new Son (30,40);
        }
        catch (wrongAge Exception e) {
            S.O.P ("Exception caught:"+e.getMessage
        }
    }
}
```

Output:

```
Father created with age : 40
Son created with age : 20
Exception caught: Father's age cannot be
 negative.
Exception caught: Son's age cannot be
greater than or equal to Father's age.
```

o/p seen

[illegible]
27/11/24

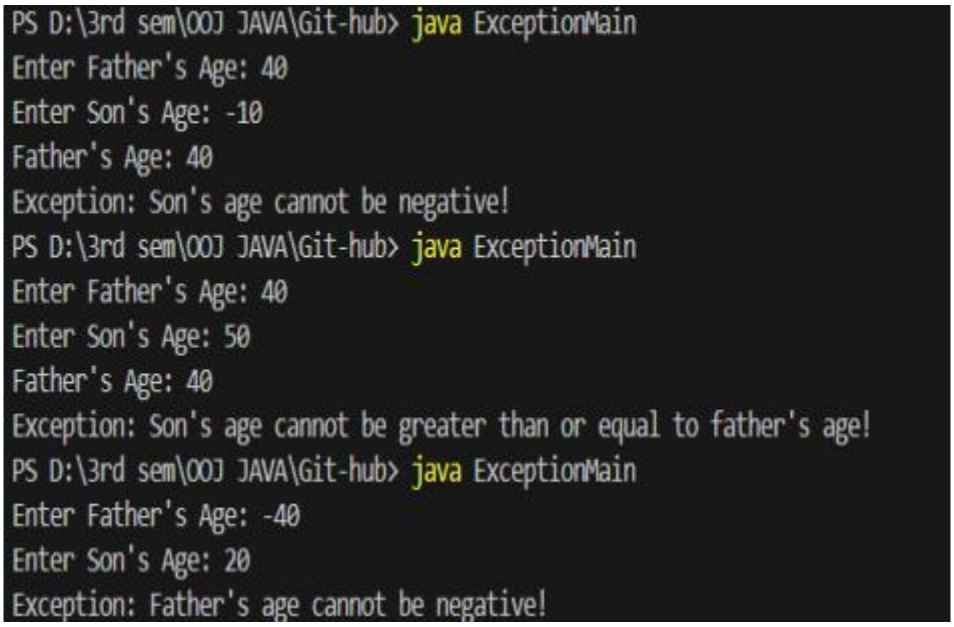


Program 7:

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's class, implement a constructor that uses both father and son's age and throws an exception if son's age is >= father's age.

```
Class WrongAgeException extends Exception{
    public WrongAgeException (string message){
        super (message);
    }
}

class Father {
    int fatherAge ;
    public Father (int fatherAge ) throws
      Wrong Age Exception {
       if (fatherAge < 0){
          throws new WrongAgeExcuption("Father
       age cannot be negative.");
       }
       this. fatherAge = fatherAge ;
    }
}
```

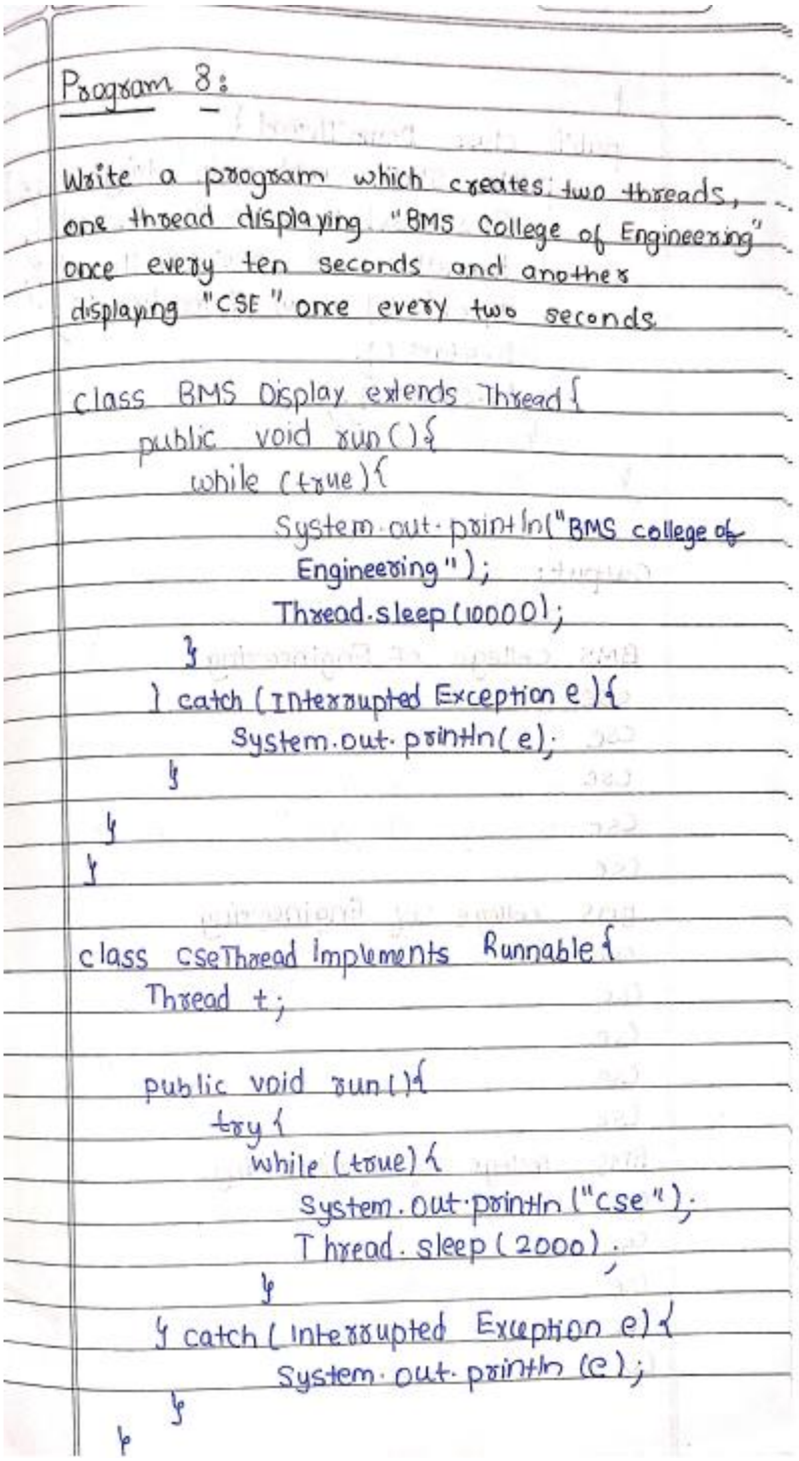
```
class WrongAgeException extends Exception {
 public WrongAgeException(String message) {
 super(message); } }
class Father {
 int fatherAge;
 public Father(int age) throws
WrongAgeException {
 if (age < 0) {
 throw new WrongAgeException("Father's age
cannot be
negative!");
 }
 this.fatherAge = age;
 System.out.println("Father's Age: " +
fatherAge); } }
class Son extends Father {
 int sonAge;
 public Son(int fatherAge, int sonAge) throws
WrongAgeException {
 super(fatherAge);
 if (sonAge < 0) {
 throw new WrongAgeException("Son's age
cannot be negative!");
 }
 if (sonAge >= fatherAge) {
 throw new WrongAgeException("Son's age
cannot be greater than
or equal to father's age!");
 }
 this.sonAge = sonAge;
 System.out.println("Son's Age: " + sonAge); } }
public class ExceptionMain {
44
 public static void main(String[] args) {
 java.util.Scanner sc = new
java.util.Scanner(System.in);
 try {
 System.out.print("Enter Father's Age: ");
 int fatherAge = sc.nextInt();
 System.out.print("Enter Son's Age: ");
 int sonAge = sc.nextInt();
 Son son = new Son(fatherAge, sonAge);
 } catch (WrongAgeException e) {
 System.out.println("Exception: " +
e.getMessage());
 } catch (Exception e) {
 System.out.println("Unexpected Exception: " +
e);} } }
```



```
PS D:\3rd sem\OOJ JAVA\Git-hub> java ExceptionMain
Enter Father's Age: 40
Enter Son's Age: -10
Father's Age: 40
Exception: Son's age cannot be negative!
PS D:\3rd sem\OOJ JAVA\Git-hub> java ExceptionMain
Enter Father's Age: 40
Enter Son's Age: 50
Father's Age: 40
Exception: Son's age cannot be greater than or equal to father's age!
PS D:\3rd sem\OOJ JAVA\Git-hub> java ExceptionMain
Enter Father's Age: -40
Enter Son's Age: 20
Exception: Father's age cannot be negative!
```

LABORATORY PROGRAM – 8

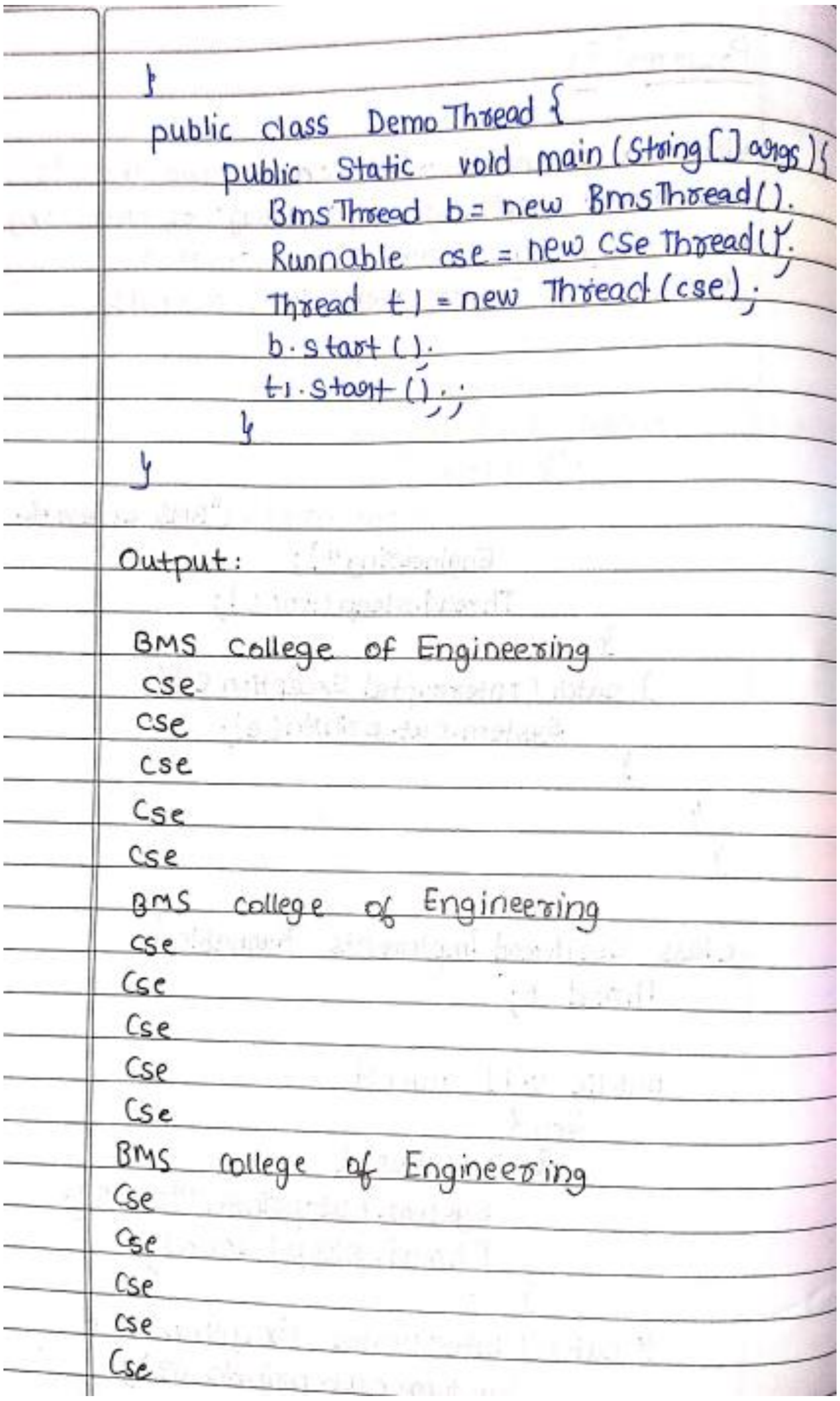
Write a program which creates two threads, one thread displaying “BMS College of Engineering” once every ten seconds and another displaying “CSE” once every two seconds.



Program 8:

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

```
class BMS Display extends Thread{
    public void run(){
        while (true){
                System.out.println("BMS college of
                 Engineering");
                Thread.sleep(10000);
        }
     } catch (InterruptedException e){
            System.out.println(e);
     }
   }
}

class CseThread implements Runnable{
    Thread t;

    public void run(){
        try{
           while (true){
                System.out.println("cse");
                Thread.sleep(2000);
           }
     } catch (Interrupted Exception e){
                System.out.println(e);
     }
  }
```



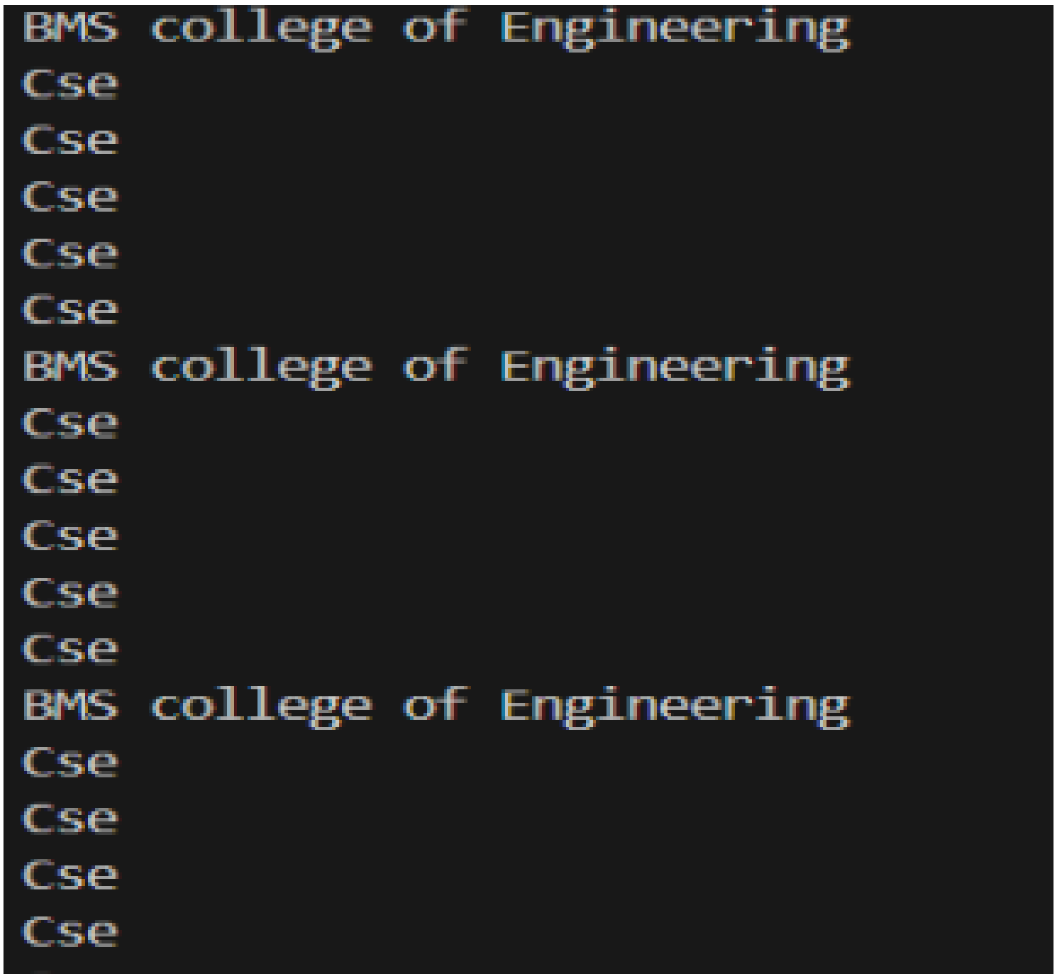
```
}
public class DemoThread {
    public Static void main(String[] args){
        BmsThread b= new BmsThread().
        Runnable cse = new CSe Thread();
        Thread t1 = new Thread(cse);
        b.start();
        t1.start();
    }
}
```

Output:

```
BMS College of Engineering
cse
cse
cse
Cse
Cse
BMS college of Engineering
cse
Cse
Cse
Cse
Cse
BMS college of Engineering
Cse
Cse
Cse
cse
Cse
```

```
class BmsThread extends Thread{
 public void run(){
 try{
 while (true) {
 System.out.println("BMS college of Engineering");
 Thread.sleep(10000);
 }
 }catch(InterruptedException e){
```

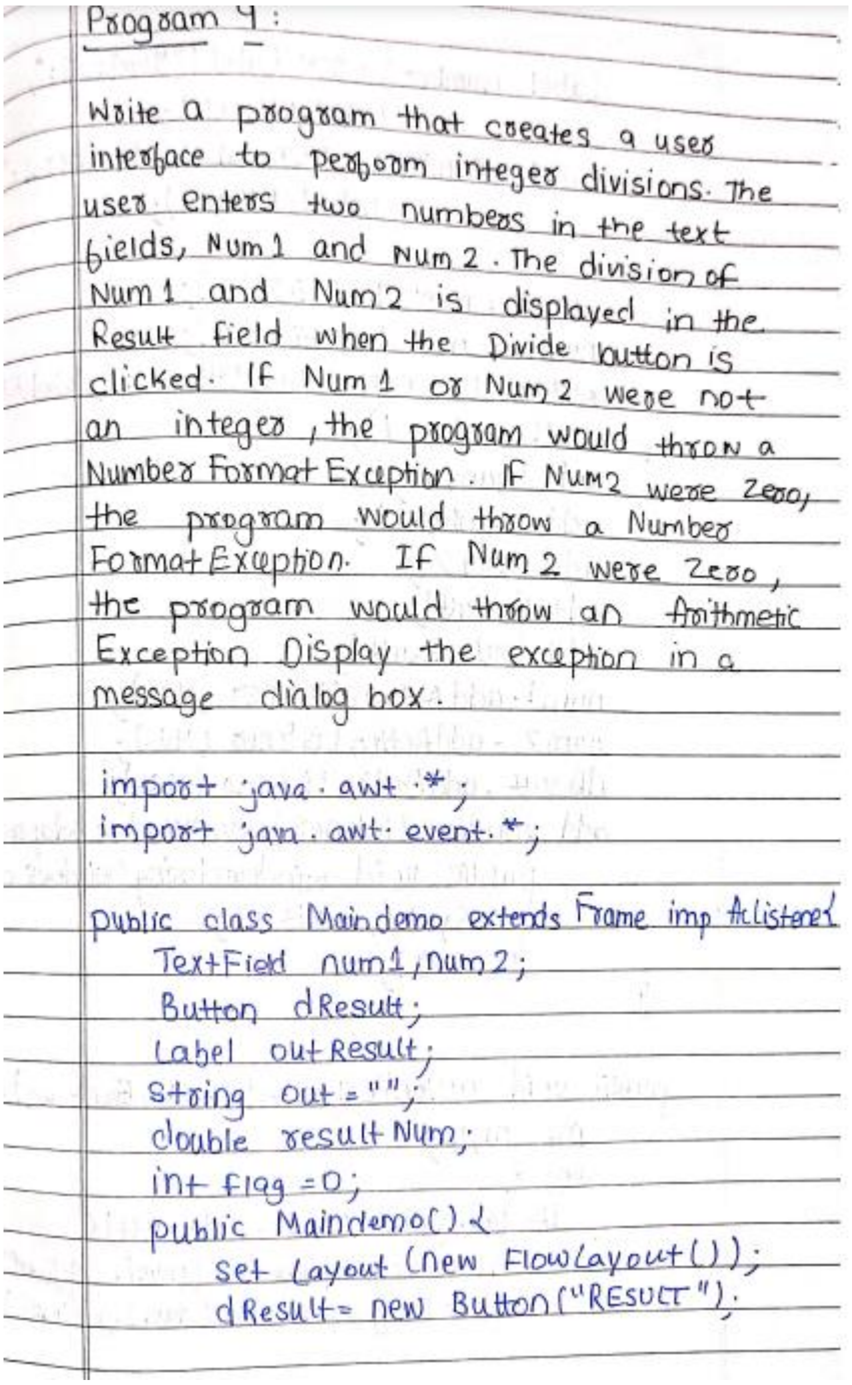
```
 System.out.println(e);
 }
 }
}
 class CseThread implements Runnable{
 Thread t;
 public void run(){
 try{
 while (true) {
 System.out.println("Cse");
 Thread.sleep(2000);
 }
 }catch(InterruptedException e){
 System.out.println(e);
 }
 }
49
}
public class DemoThread {
 public static void main(String[] args) {
 BmsThread b=new BmsThread();
 Runnable cse=new CseThread();
 Thread t1=new Thread(cse);
 b.start();
 t1.start();;
 }
}
```



```
BMS college of Engineering
Cse
Cse
Cse
Cse
Cse
BMS college of Engineering
Cse
Cse
Cse
Cse
Cse
BMS college of Engineering
Cse
Cse
Cse
Cse
```

LABORATORY PROGRAM - 9

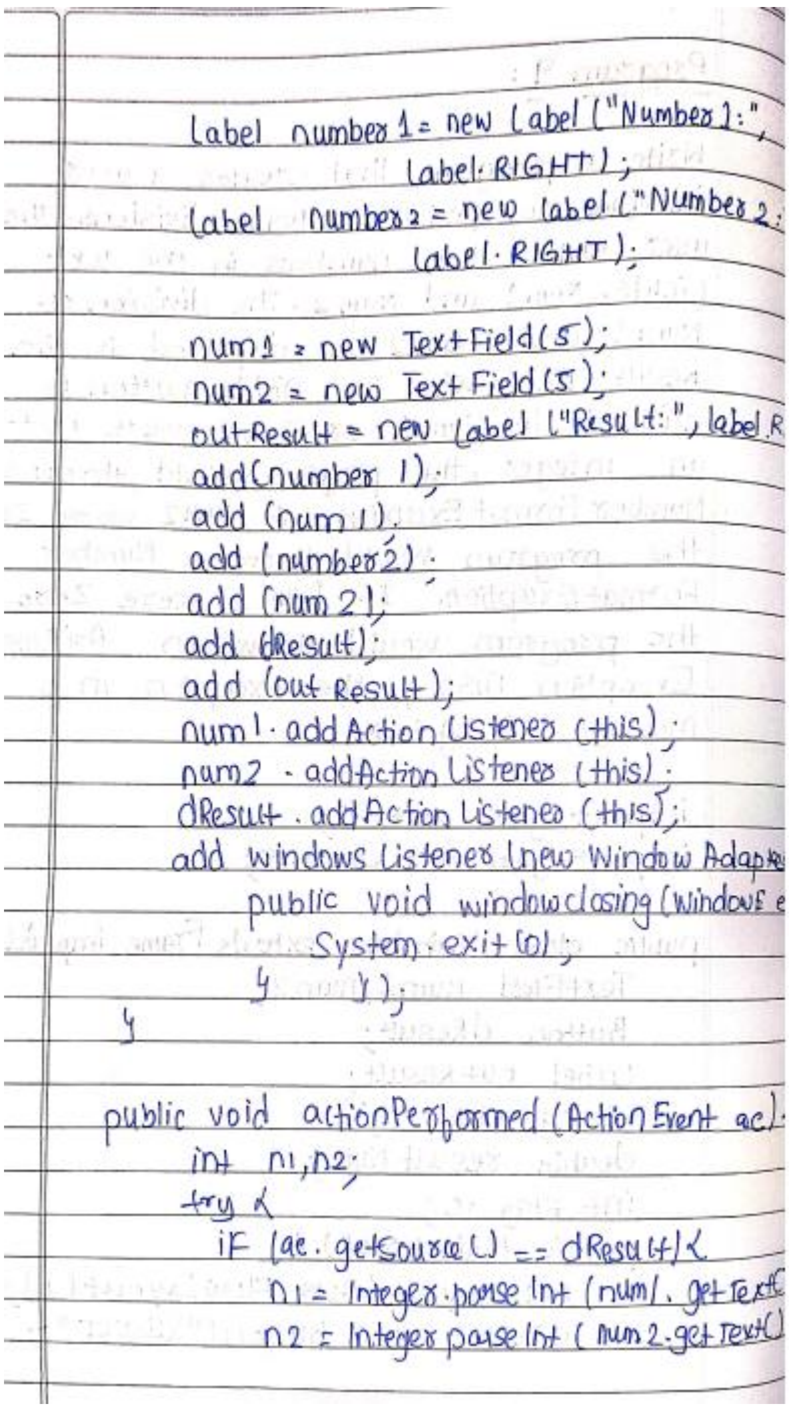
Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException.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



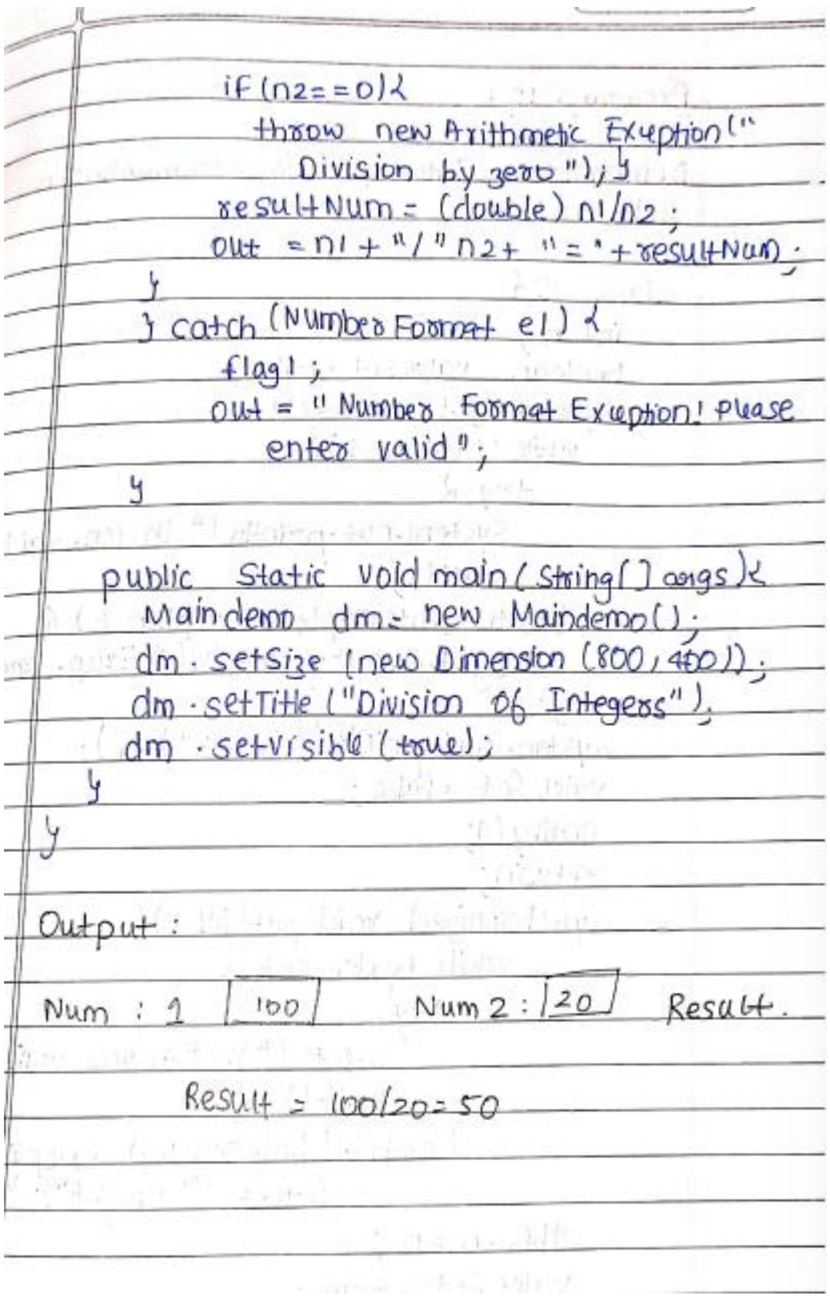
Program 9 :

Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num 1 and Num 2. The division of Num 1 and Num 2 is displayed in the Result field when the Divide button is clicked. If Num 1 or Num 2 were not an integer, the program would thron a Number Format Exception. If Num2 were zero, the program would throw a Number FormatException. If Num 2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

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

public class Maindemo extends Frame imp AcListener{
    TextField num1,num2;
    Button dResult;
    Label outResult;
    String out="";
    double resultNum;
    int flag=0;
    public Maindemo(){
        set Layout(new FlowLayout());
        dResult= new Button("RESULT");
```



```
        label number1 = new label("Number1:",
                        label.RIGHT);
        label number2 = new label("Number 2:
                        label.RIGHT);

        num1 = new TextField(5);
        num2 = new Text Field(5);
        outResult = new label ("Result:", label.R
        add(number 1);
        add (num1);
        add (number2);
        add (num 2);
        add (dResult);
        add (out Result);
        num1.add Action Listener (this);
        num2 .addAction Listener (this);
        dResult .add Action Listener (this);
        add windows Listener (new Window Adapte
            public void windowclosing (Windows e
                System.exit(0);
            }});
    }

public void actionPerformed (Action Event ac)
        int n1,n2;
        try {
          if (ac.getSource() == dResult){
            n1 = Integer.parse Int (num1. getText
            n2 = Integer parse Int ( num 2.get Text()
```



```
        if(n2==0){
          throw new Arithmetic Exeption("
             Division by zero");}
        resultNum= (double) n1/n2;
        out =n1+"/"n2+ "="+resultNum;
   }
   } catch (NumberFormat e1) {
        flag1;
        out = "Number Format Exception! Please
             enter valid";
   }
  public Static void main(String[] args){
   Maindemo dm= new Maindemo();
   dm.setSize (new Dimension (800,400));
   dm.setTitle ("Division of Integers");
   dm.setvisible(true);
  }
}
```

Output :

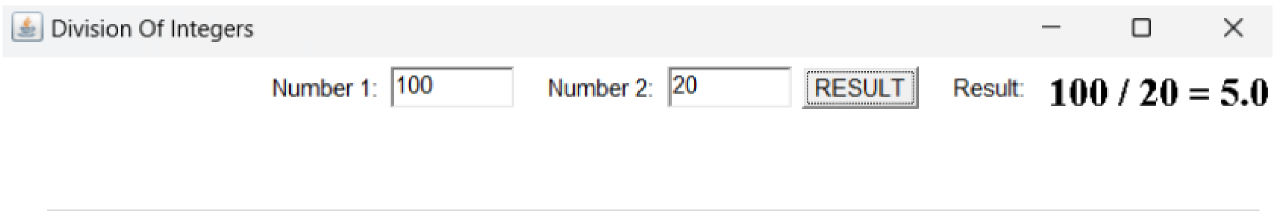
```
Num : 1 [100]        Num 2 : [20]     Result.

        Result = 100/20=50
```

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

public class DivisionMain1 extends Frame implements ActionListener
{
        TextField
        num1,num2;
        Button dResult;
        Label
        outResult;
        String
```

```
out="";
double
resultNum;
int flag=0;

public DivisionMain1()
{
        setLayout(new FlowLayout());

        dResult = new Button("RESULT");
        Label   number1   =   new   Label("Number
        1:",Label.RIGHT); Label number2 = new
        Label("Number           2:",Label.RIGHT);
        num1=new TextField(5);
        num2=new TextField(5);
        outResult = new Label("Result:",Label.RIGHT);

        add(number1
        );
        add(num1);
        add(number2
        );
        add(num2);
        add(dResult);
        add(outResul
        t);

        num1.addActionListener(this);
        num2.addActionListener(this);
        dResult.addActionListener(this);
        addWindowListener(new
        WindowAdapter()
        {
                public void windowClosing(WindowEvent we)
                {
                        System.exit(0);
                }
        });
}
```

```
public void actionPerformed(ActionEvent ae)
{
        int
        n1,n2;
        try
        {
                if (ae.getSource() == dResult)
                {
                        n1=Integer.parseInt(num1.getText());
                        n2=Integer.parseInt(num2.getText());

                        /*if(n2==0)
                                throw new
                        ArithmeticException();*/ out=n1+"
                        "+n2+" ";
                        resultNum=n1/n2;
                        out+=String.valueOf(result
                        Num); repaint();

                }
        }
        catch(NumberFormatException e1)
        {
                flag=1;
                out="Number Format Exception!
                "+e1; repaint();
        }
        catch(ArithmeticException e2)
        {
                flag=1;
                out="Divide by 0 Exception!
                "+e2; repaint();
        }

}
public void paint(Graphics g)
{
        if(flag==0)
        g.drawString(out,outResult.getX()+outResult.getWidth(),outRes
        ult.getY()+outResult. getHeight()-8);
        else
```

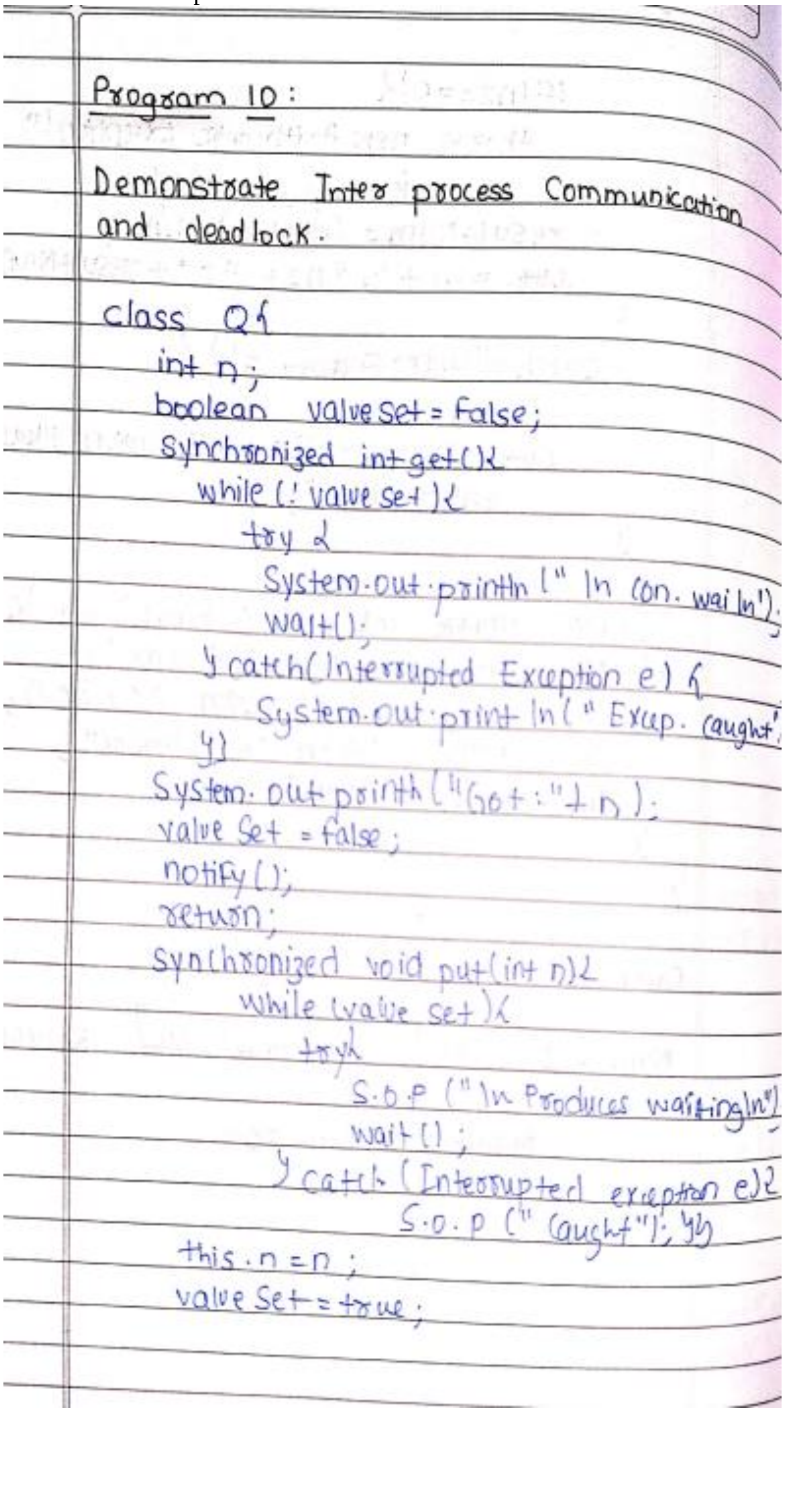
```
                g.drawString(out,10
                0,200); flag=0;
        }
```



```
100 / 20 = 5.0
```

LABORATORY PROGRAM – 10

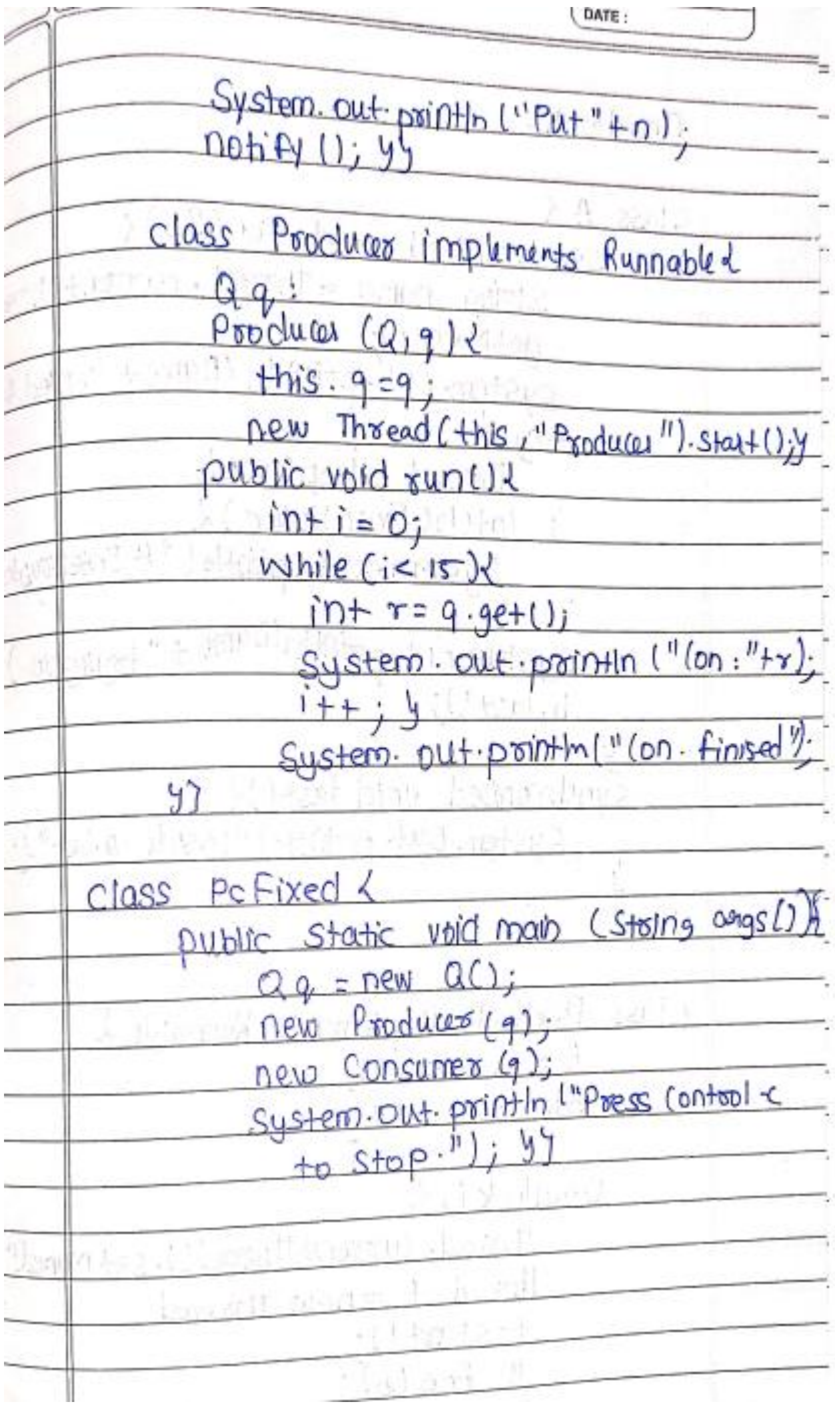
Demonstrate Interprocess communication and deadlock



Program 10:

Demonstrate Inter process Communication and deadlock.

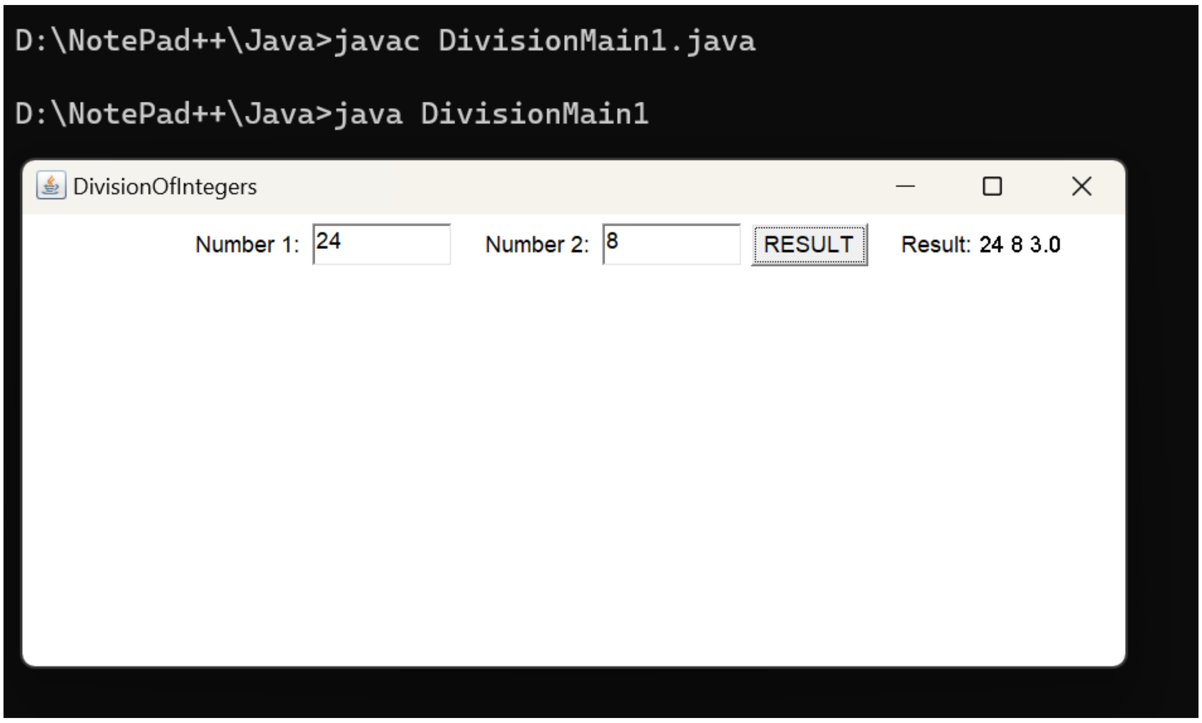
```
class Q{
   int n;
   boolean valueSet = False;
   synchronized int get(){
      while (! value set){
         try {
            System.out.println(" \n Con. wai\n");
            wait();
      } catch(Interrupted Exception e) {
            System.out.print ln(" Excp. Caught");
      }}
   System.out.printh("Got:"+n);
   value Set = false;
   notify();
   return;
   synchronized void put(int n){
         while (value set){
            try{
                  S.O.P ("\n Produces waiting\n");
                  wait();
            } catch (Interupted exception e){
                     S.O.P (" Caught"); }}
      this.n = n;
      value Set = true;
```


DATE :

```
System.out.println("Put"+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){
      int r= q.get();
      System.out.println("Con:"+r);
      i++; }
    System.out.println("Con. finised");
}}

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."); }}
```

```
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("\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;
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 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

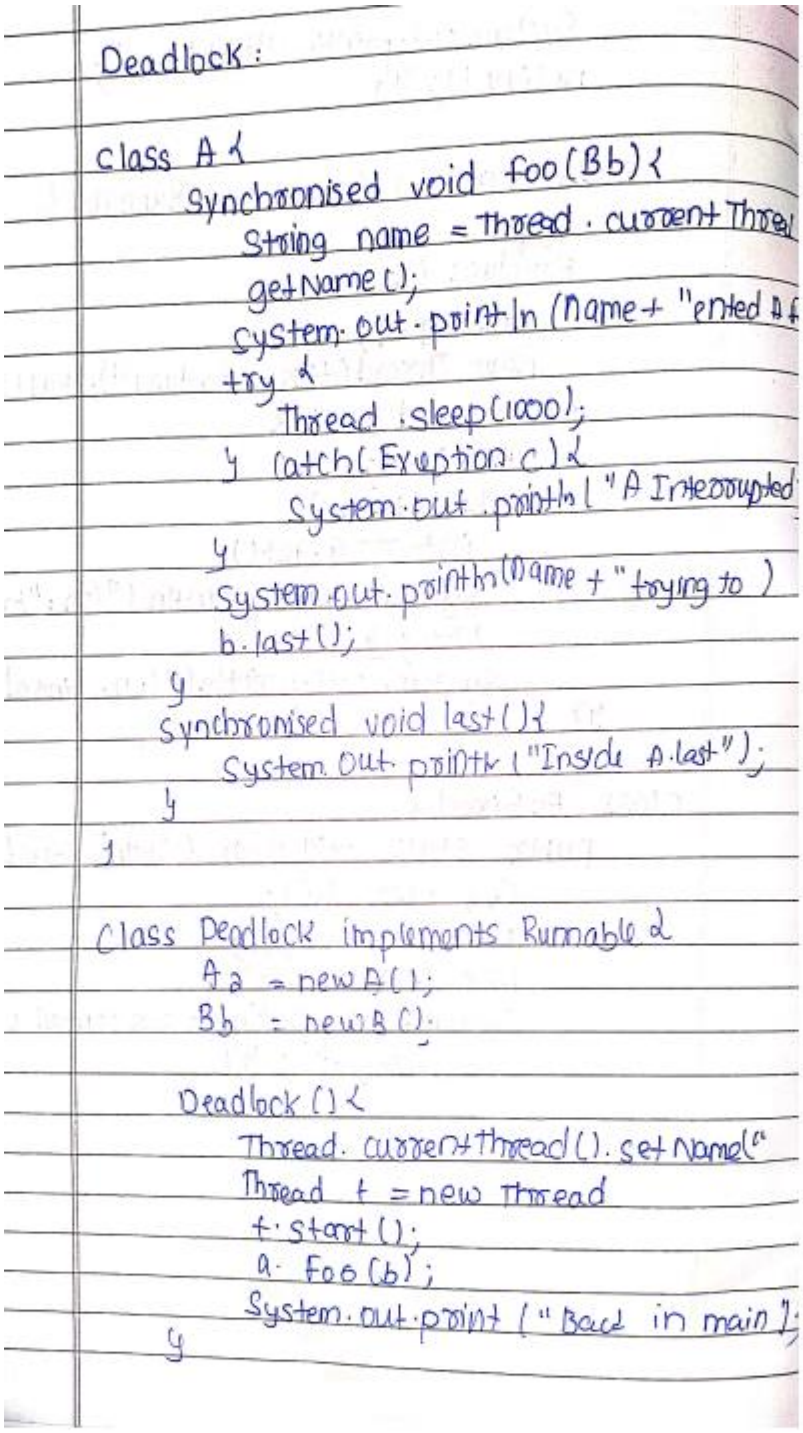


```
D:\NotePad++\Java>javac DivisionMain1.java

D:\NotePad++\Java>java DivisionMain1
```

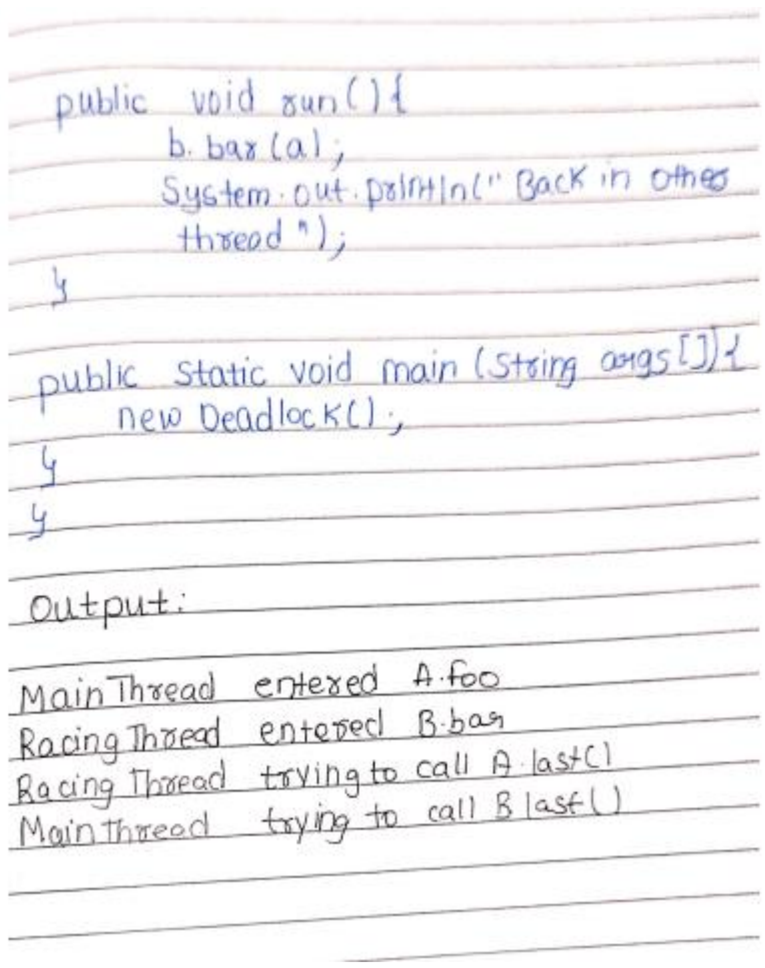
DivisionOfIntegers

Number 1: 24 Number 2: 8 RESULT Result: 24 8 3.0

ii. Demonstration of deadlock



```
Deadlock:

class A {
    Synchronised void foo(Bb){
        String name = thread . currentThrea
        getName();
        System.out.println(name + "ented A f
        try {
            Thread.sleep(1000);
        } catch(Exeption c){
            System.out.println("A Interrupted
        }
        System.out.println(name + "trying to )
        b.last();
    }
    synchronised void last(){
        System.out.println("Inside A.last");
    }
}

class Deadlock implements Runnable {
    A a = new A();
    B b = new B();

    Deadlock(){
        Thread.currentthread().setName("
        Thread t = new Thread
        t.start();
        a.Foo(b);
        System.out.print("Back in main");
    }
```

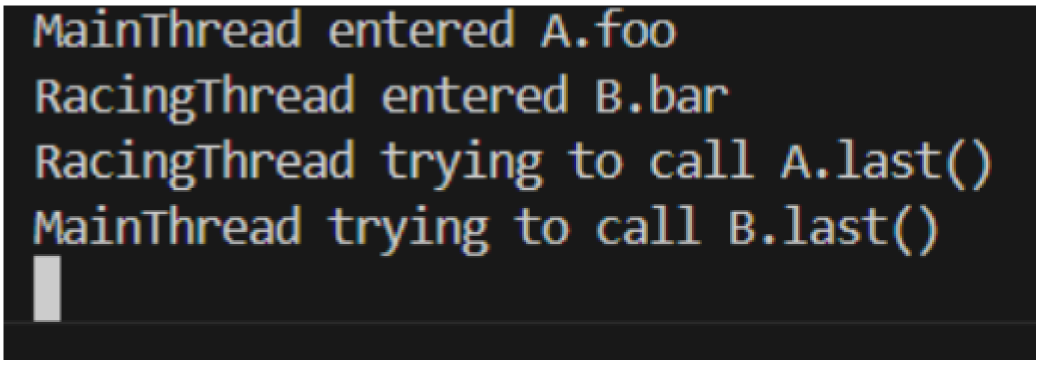


```
public void run(){
        b.bar(a);
        System.out.println(" Back in other
        thread ");
}

public Static void main (String args[]){
    new Deadlock();
}
}
```

Output:

```
MainThread   entered  A.foo
RacingThread  entered  B.bar
RacingThread  trying to call A.last()
MainThread   trying to call B last()
```

```
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 A.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); // get lock on a in this thread.
     System.out.println("Back in main thread");
   }
 public void run() { b.bar(a); // get lock on b in other thread.
  System.out.println("Back in other thread");
  }
public static void main(String args[]) { new Deadlock(); }
```



```
MainThread entered A.foo
RacingThread entered B.bar
RacingThread trying to call A.last()
MainThread trying to call B.last()
```