VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

SHAIKH UZAIR AHMED
(1BM23CS307)

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

CERTIFICATE

This is to certify that the Lab work entitled "**OBJECT ORIENTED JAVA PROGRAMMING**" carried out by SHAIKH UZAIR AHMED**(1BM23CS307),** 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

Associate Professor,
Department of CSE,
BMSCE, Bengaluru

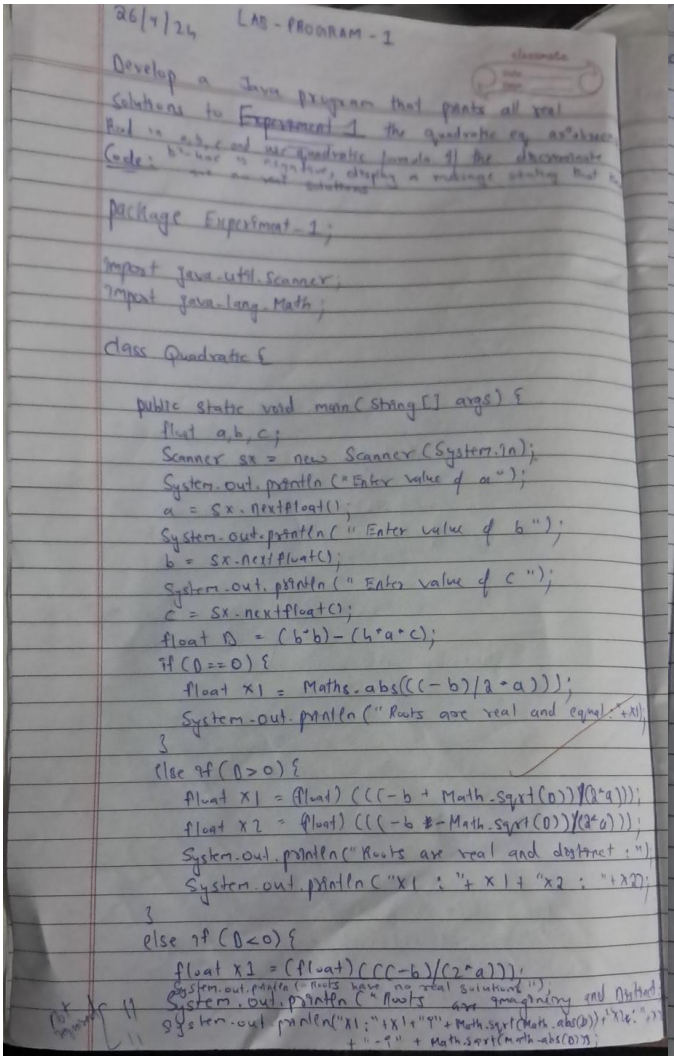
Dr. Kavitha Sooda

Professor and Head,
Department of CSE
BMSCE, Bengaluru

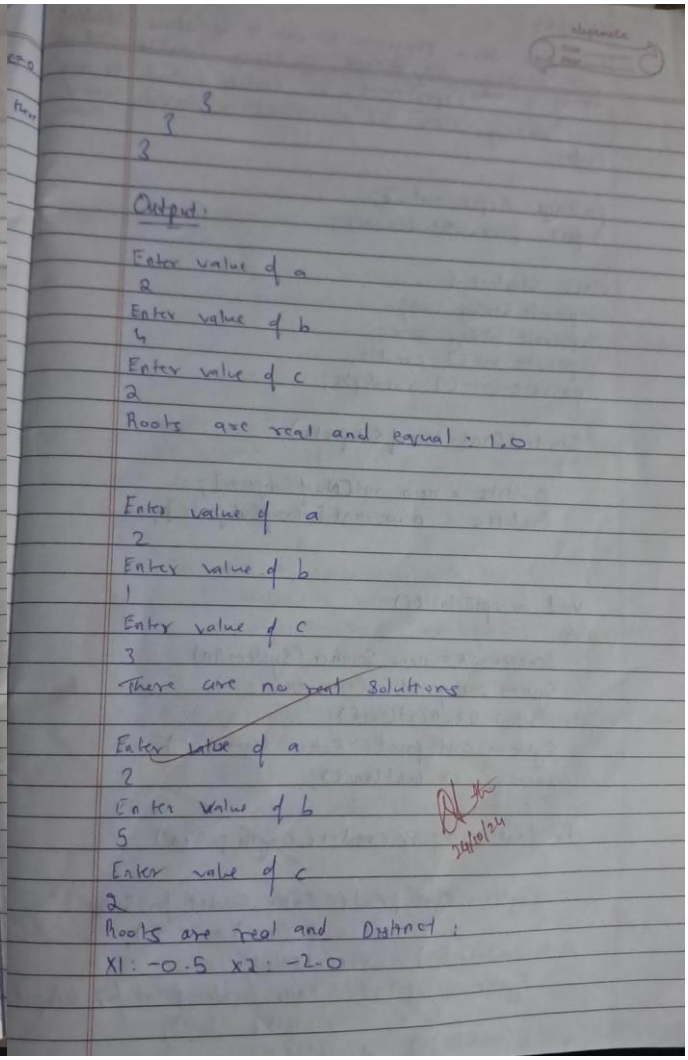
INDEX

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.



```
26/9/24  LAB - PROGRAM - 1

Develop a Java program that prints all real
solutions to Experiment 1 the quadratic eq ax²+bx+c=0
Aim : a,b,c and use quadratic formula if the discriminate
Code: b²-4ac is negative, display a message stating that
there are no real solutions

package Experiment_1;

import java.util.Scanner;
import java.lang.Math;

class Quadratic {

  public static void main(String[] args) {
    float a,b,c;
    Scanner sx = new Scanner(System.in);
    System.out.println("Enter value of a");
    a = sx.nextFloat();
    System.out.println("Enter value of b");
    b = sx.nextFloat();
    System.out.println("Enter value of c");
    c = sx.nextFloat();
    float D = (b*b)-(4*a*c);
    If (D==0) {
      float X1 = Maths.abs(((-b)/2*a)));
      System.out.println("Roots are real and equal:"+X1);
    }
    else if (D>0) {
      float X1 = (float) (((-b + Math.sqrt(D))/(2*a)));
      float X2 = (float) (((-b - Math.sqrt(D))/(2*a)));
      System.out.println("Roots are real and distinct:");
      System.out.println("X1 : "+ X1 + "X2 : "+X2);
    }
    else if (D<0) {
      float X1 = (float)(((-b)/(2*a)));
      System.out.println("Roots have no real solutions");
      System.out.println("Roots are imaginary and distinct:");
      System.out.println("X1:"+X1+"i"+Math.sqrt(Math.abs(D))+"X2:"
          +X1+"-i"+Math.sqrt(Math.abs(D)));
    }
  }
}
```



```
Output:

Enter value of a
2
Enter value of b
4
Enter value of c
2
Roots are real and equal : 1.0

Enter value of a
2
Enter value of b
1
Enter value of c
3
There are no real solutions

Enter value of a
2
Enter value of b
5
Enter value of c
2
Roots are real and Distinct :
X1 : -0.5  X2 : -2.0
```

Code:

```
package Experiment_1;

import java.util.Scanner;
import java.lang.Math;
class Quadratic {
   public static void main(String[] args) {
      float a,b,c;
      Scanner sx = new Scanner(System.in);
      System.out.println("Enter value of a");
      a = sx.nextFloat();
      System.out.println("Enter value of b");
      b = sx.nextFloat();
      System.out.println("Enter value of c");
      c = sx.nextFloat();
      float D = (b*b)-(4*a*c);
      if (D==0) {
         float X1 = Math.abs(((-b)/(2*a)));
         System.out.println("Roots are real and equal : " + X1);
      }
```

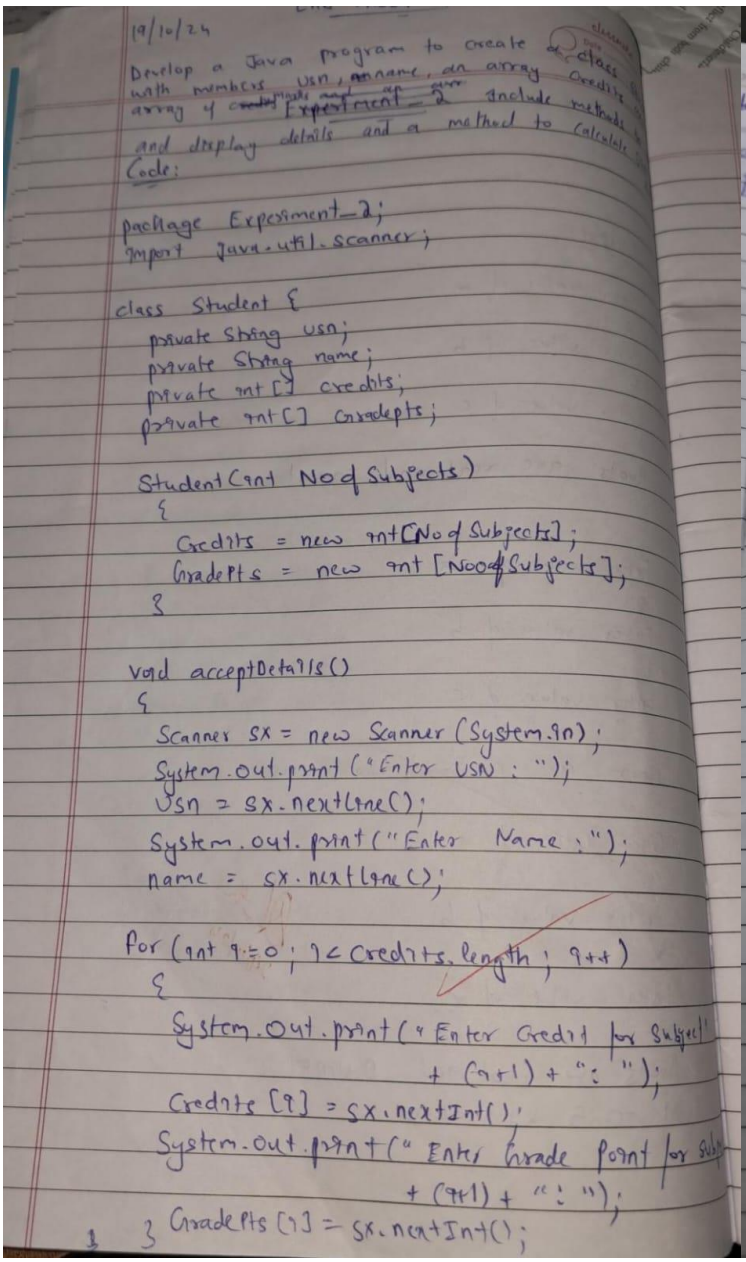
```
        else if(D>0){
            float X1 = (float) (((-b+Math.sqrt(D))/(2*a)));
            float X2 = (float) (((-b-Math.sqrt(D))/(2*a)));
            System.out.println("Roots are real and Distinct : ");
            System.out.println("X1 : " + X1+"X2 : "+X2);
        }
        else if(D<0){
            float X1 = (float) (((-b)/(2*a)));
            System.err.println("There are no real solutions");
            //System.out.println("Roots are Imaginary and Distinct : ");
            //System.out.println("X1 : " + X1+"i"+Math.sqrt(Math.abs(D))+"X2 : "+X1+"-
i"+Math.sqrt(Math.abs(D)));
        }

    }
}
```

```
PS C:\Users\uzair\OneDrive\Desktop\java> java Experiment_1.Quadratic
Enter value of a
2
Enter value of b
1
Enter value of c
3
There are no real solutions
PS C:\Users\uzair\OneDrive\Desktop\java> java Experiment_1.Quadratic
Enter value of a
2
Enter value of b
4
Enter value of c
2
Roots are real and equal : 1.0
PS C:\Users\uzair\OneDrive\Desktop\java> java Experiment_1.Quadratic
Enter value of a
2
Enter value of b
5
Enter value of c
2
Roots are real and Distinct :
X1 : -0.5X2 : -2.0
```

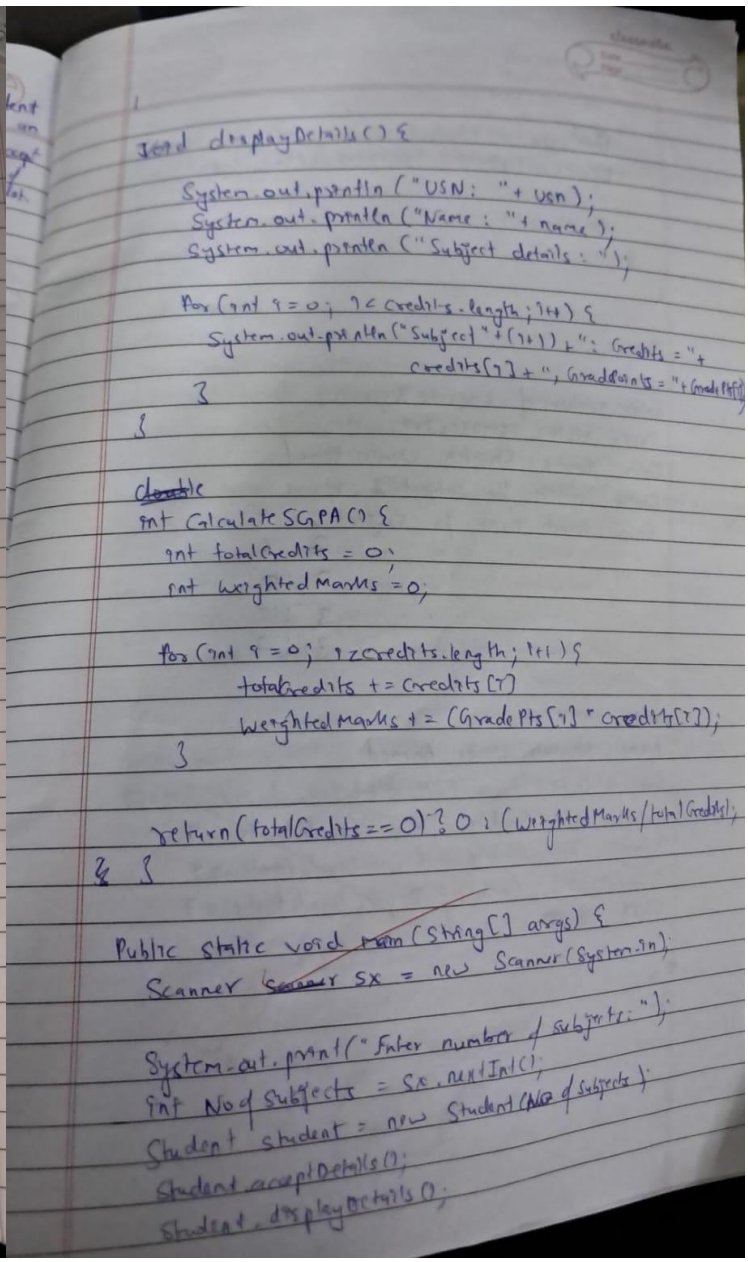
Output

Program 2

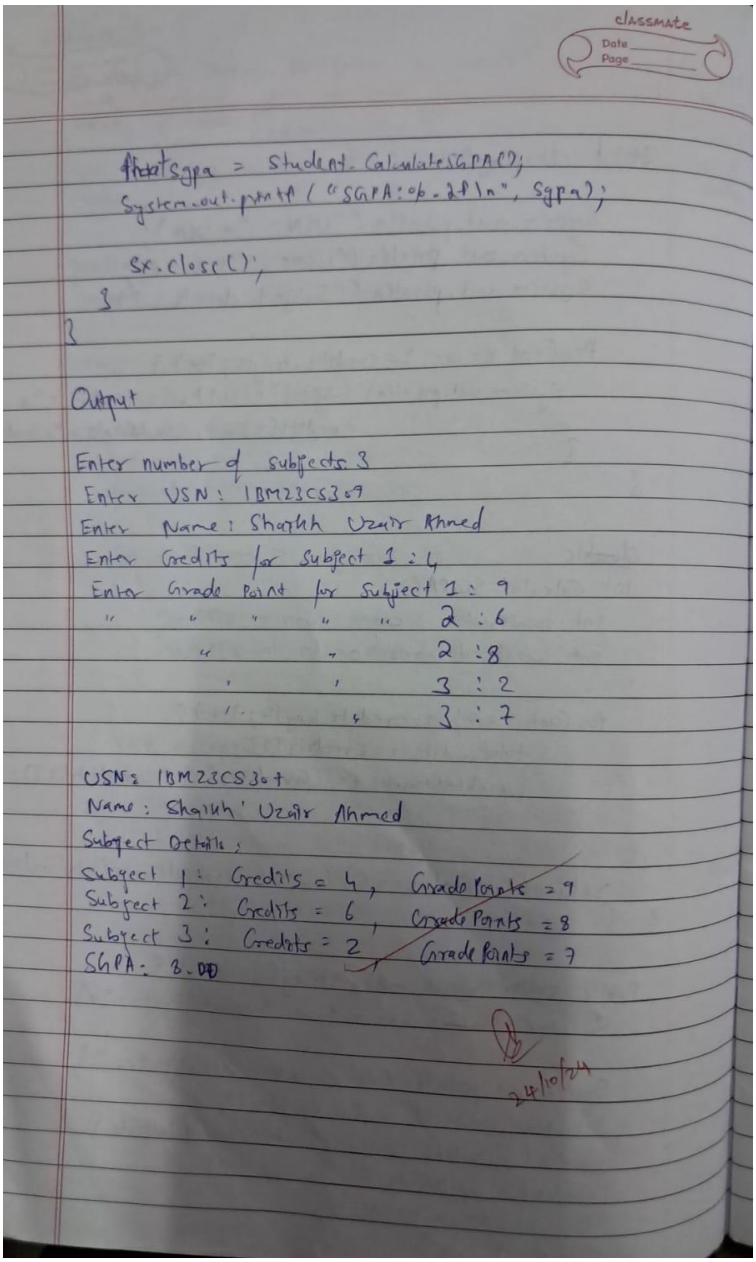
Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.



```
19/10/24
Develop a Java program to create a class Student with members usn, name, an array credits and an array of marks. Include methods to accept and display details and a method to calculate SGPA of a student.
Code:

package Experiment_2;
import java.util.Scanner;

class Student {
    private String usn;
    private String name;
    private int[] credits;
    private int[] gradepts;

    Student(int No of Subjects)
    {
        Credits = new int[No of Subjects];
        Gradepts = new int [No of Subjects];
    }

    void acceptDetails()
    {
        Scanner sx = new Scanner (System.in);
        System.out.print ("Enter USN : ");
        Usn = sx.nextLine();
        System.out.print("Enter Name :");
        name = sx.nextLine();

        for (int i=0; i<Credits.length; i++)
        {
            System.out.print("Enter Credit for Subject"
                    + (i+1) + ": ");
            Credits[i] = sx.nextInt();
            System.out.print(" Enter Grade Point for Sub
                    + (i+1) + ": ");
        } Gradepts[i] = sx.nextInt();
```



```
void displayDetails() {

    System.out.println ("USN: " + usn);
    System.out.println ("Name : " + name);
    System.out.println ("Subject details : ");

    for (int i = 0; i < credits.length; i++) {
        System.out.println ("Subject " + (i+1) + ": Credits = "
                + credits[i] + ", Gradepoints = " + Gradepts[i]);
    }
}

double
int CalculateSGPA() {
    int totalCredits = 0;
    int weightedMarks = 0;

    for (int i = 0; i < credits.length; i++) {
        totalCredits += Credits[i];
        weightedMarks += (GradePts[i] * Credits[i]);
    }

    return (totalCredits == 0) ? 0 : (weightedMarks / totalCredits);
} }

Public static void main (String[] args) {
    Scanner sx = new Scanner (System.in);

    System.out.print(" Enter number of subjects: ");
    int No of Subjects = sx.nextInt();
    Student student = new Student(No of Subjects);
    Student.acceptDetails();
    Student.displayDetails();
```



```
float Sgpa = Student.CalculateSGPA();
System.out.printf("SGPA: %.2f\n", Sgpa);

Sx.close();
}
}
```

Output

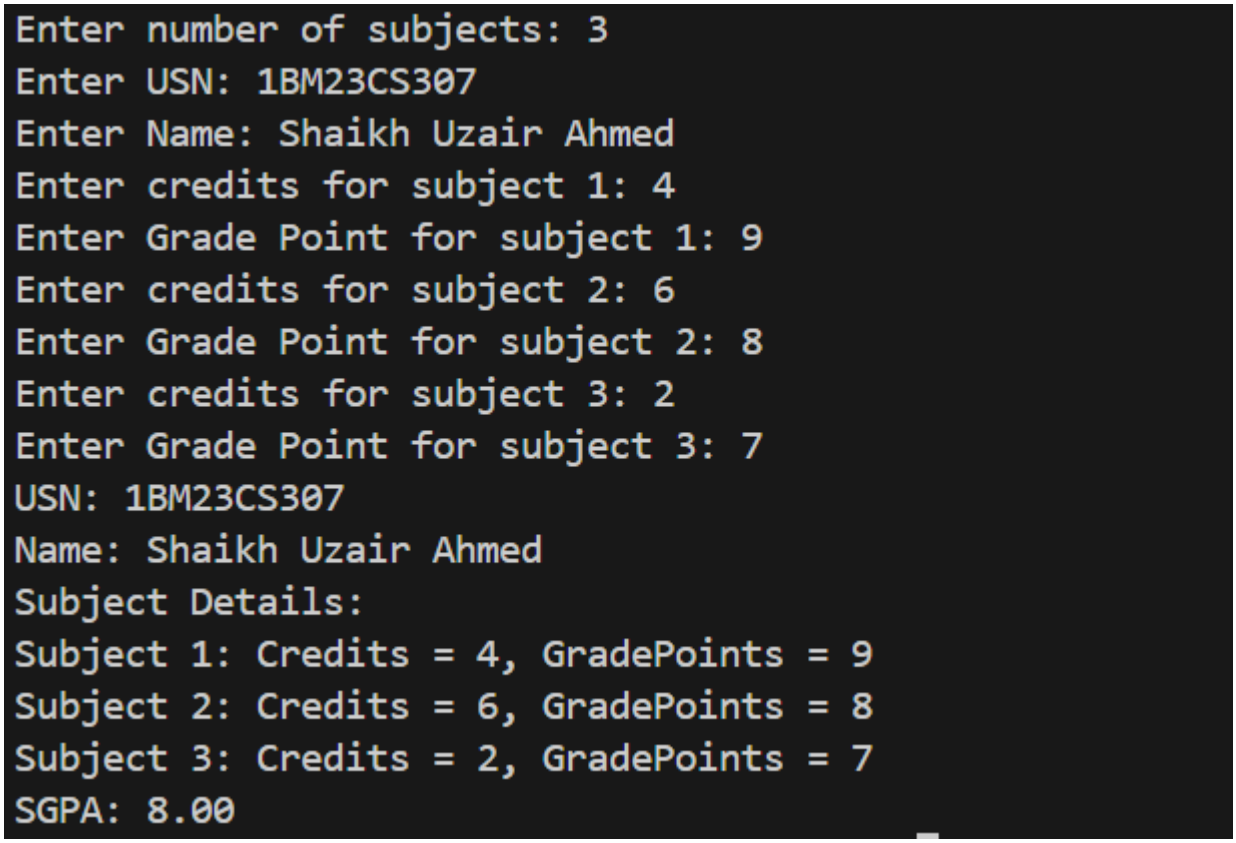
```
Enter number of subjects: 3
Enter USN: 1BM23CS309
Enter Name: Shaikh Uzair Ahmed
Enter Credits for Subject 1: 4
Enter Grade Point for Subject 1: 9
" " " " " 2: 6
" " " 2: 8
" " " 3: 2
" " " 3: 7

USN: 1BM23CS309
Name: Shaikh Uzair Ahmed
Subject Details:
Subject 1: Credits = 4, Grade Points = 9
Subject 2: Credits = 6, Grade Points = 8
Subject 3: Credits = 2, Grade Points = 7
SGPA: 8.00
```

24/10/24

Code:

```
package Experiment_2;
import java.util.Scanner;

class Student {
    private String usn;
    private String name;
    private int[] credits;
    private int[] GradePts;

    Student(int numberOfSubjects) {
        credits = new int[numberOfSubjects];
        GradePts = new int[numberOfSubjects];
    }

    void acceptDetails() {
        Scanner sx = new Scanner(System.in);

        System.out.print("Enter USN: ");
        usn = sx.nextLine();
```

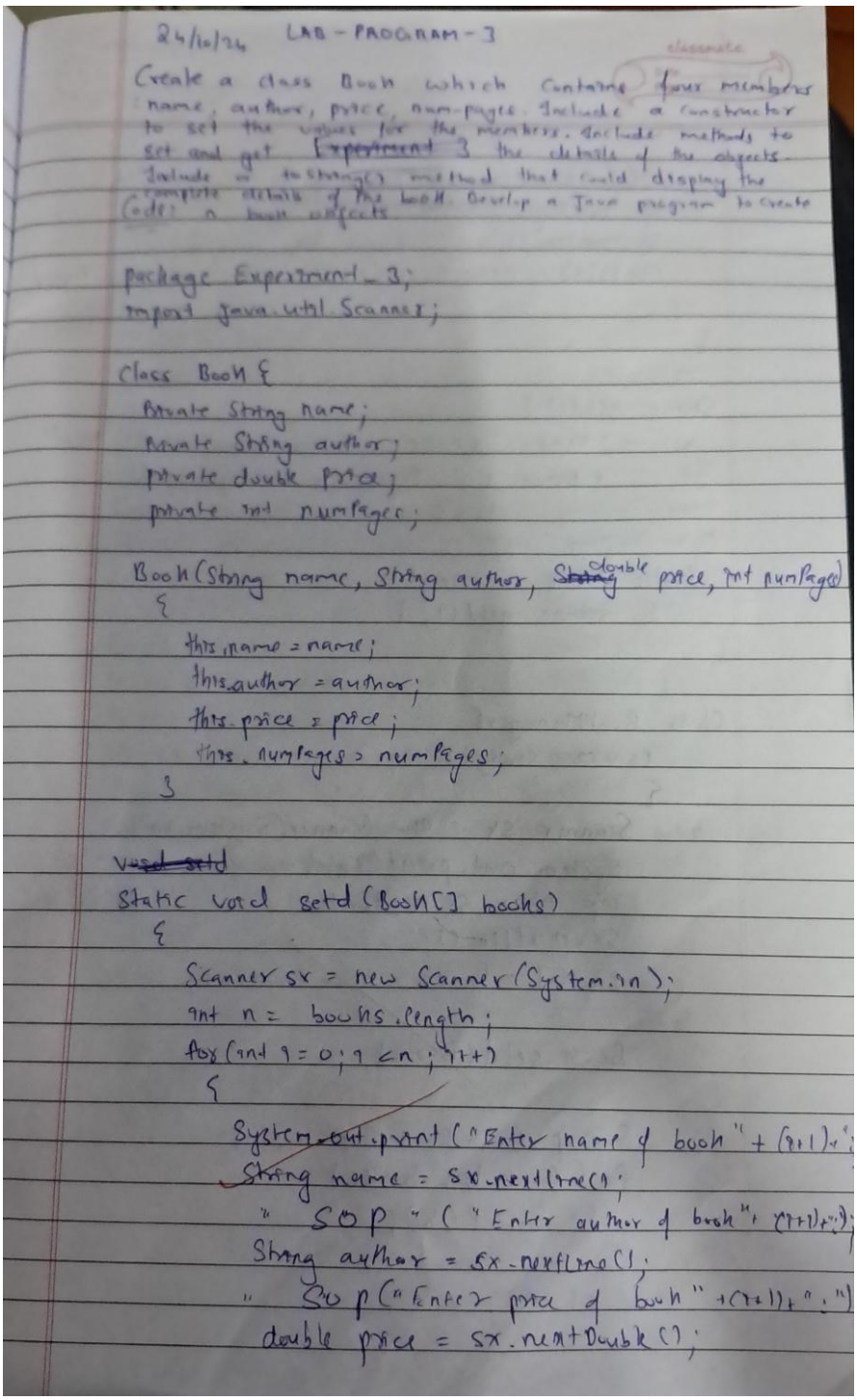
```
        System.out.print("Enter Name: ");
        name = sx.nextLine();

        for (int i = 0; i < credits.length; i++) {
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
            credits[i] = sx.nextInt();

            System.out.print("Enter Grade Point for subject " + (i + 1) + ": ");
            GradePts[i] = sx.nextInt();
        }
    }

    void displayDetails() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);

        System.out.println("Subject Details:");
        for (int i = 0; i < credits.length; i++) {
            System.out.println("Subject " + (i + 1) + ": Credits = " + credits[i] + ", GradePoints = " +
GradePts[i]);
        }
    }

    int calculateSGPA() {
        int totalCredits = 0;
        int weightedMarks = 0;

        for (int i = 0; i < credits.length; i++) {
            totalCredits += credits[i];
            weightedMarks += (GradePts[i] * credits[i]);
        }

        return (totalCredits == 0) ? 0 : (weightedMarks / totalCredits);
    }

    public static void main(String[] args) {
        Scanner sx = new Scanner(System.in);

        System.out.print("Enter number of subjects: ");
        int numberOfSubjects = sx.nextInt();

        Student student = new Student(numberOfSubjects);
        student.acceptDetails();
        student.displayDetails();

        float sgpa = student.calculateSGPA();
        System.out.printf("SGPA: %.2f\n", sgpa);

        sx.close();
    }
}
```


Enter number of subjects: 3
Enter USN: 1BM23CS307
Enter Name: Shaikh Uzair Ahmed
Enter credits for subject 1: 4
Enter Grade Point for subject 1: 9
Enter credits for subject 2: 6
Enter Grade Point for subject 2: 8
Enter credits for subject 3: 2
Enter Grade Point for subject 3: 7
USN: 1BM23CS307
Name: Shaikh Uzair Ahmed
Subject Details:
Subject 1: Credits = 4, GradePoints = 9
Subject 2: Credits = 6, GradePoints = 8
Subject 3: Credits = 2, GradePoints = 7
SGPA: 8.00

Output

Program 3

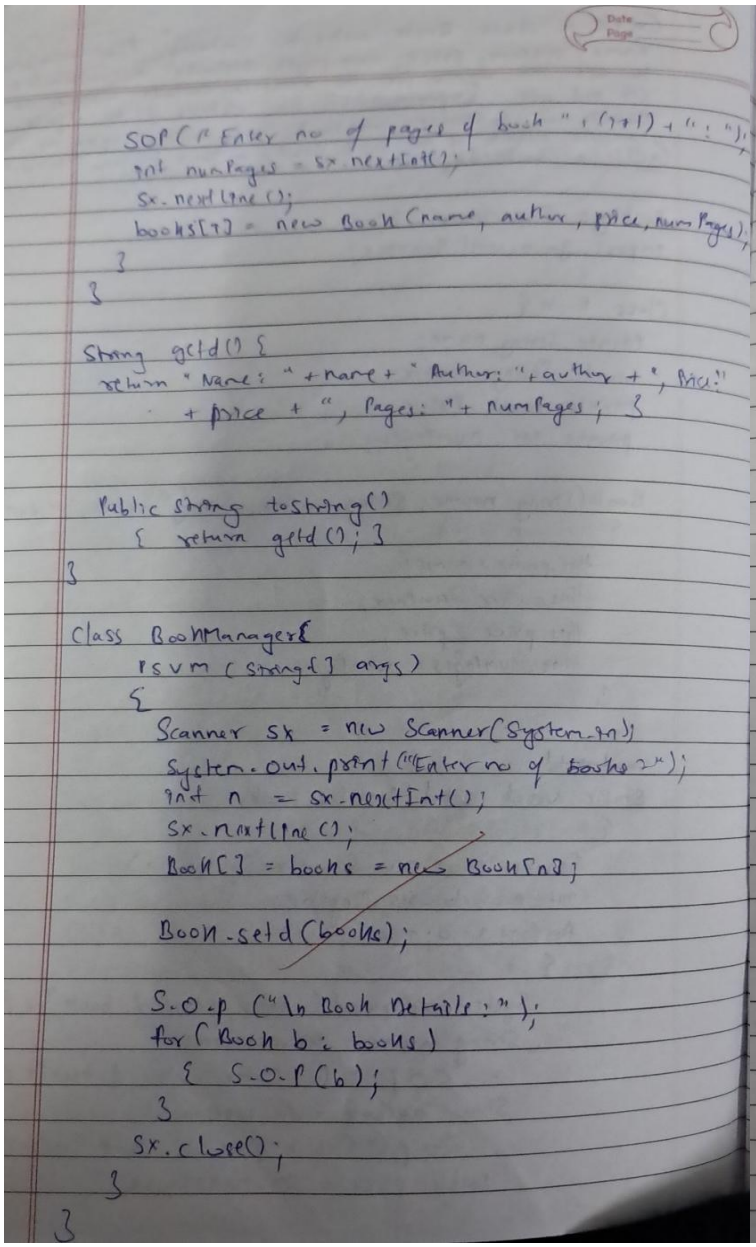
Create a class Book which contains four members: name, author, price, num_pages. Include constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString( ) method that could display the complete details of the book. Develop a Java program to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



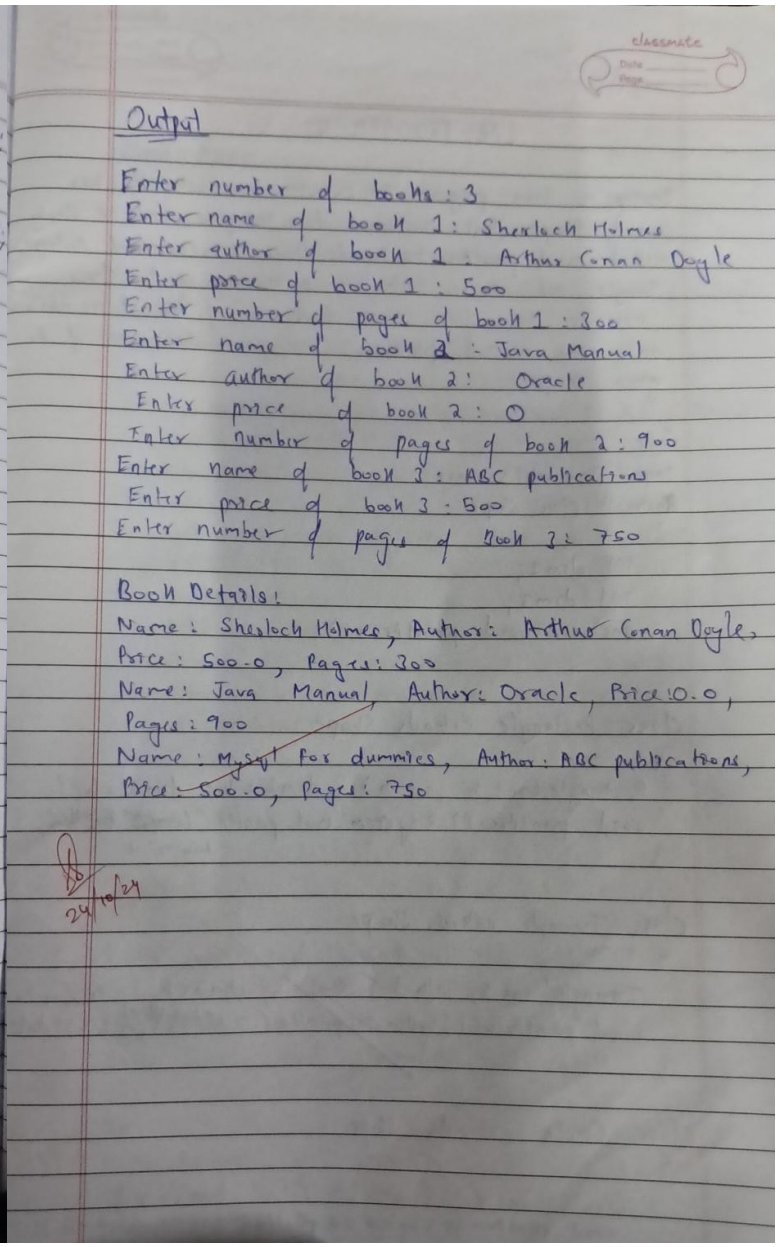
24/10/24 LAB - PROGRAM - 3

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

```
package Experiment_3;
import java.util.Scanner;

class Book {
    private String name;
    private String author;
    private double price;
    private int numPages;

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

    static void setd(Book[] books)
    {
        Scanner sx = new Scanner(System.in);
        int n = books.length;
        for(int i = 0; i < n; i++)
        {
            System.out.print("Enter name of book " + (i+1) + ": ");
            String name = sx.nextLine();
            System.out.print("Enter author of book " + (i+1) + ": ");
            String author = sx.nextLine();
            System.out.print("Enter price of book " + (i+1) + ": ");
            double price = sx.nextDouble();
```



```
SOP("Enter no of pages of book " + (i+1) + ": ");
int numPages = sx.nextInt();
sx.nextLine();
books[i] = new Book(name, author, price, numPages);
}
}

String getId() {
return "Name: " + name + " Author: " + author + ", Price: "
+ price + ", Pages: " + numPages; }

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

Class BookManager{
psvm (String[] args)
{
Scanner sx = new Scanner(System.in);
System.out.print("Enter no of books :");
int n = sx.nextInt();
sx.nextLine();
Book[] = books = new Book[n];

Book.setd(books);

S.O.p ("\n Book Details:");
for(Book b : books)
{ S.O.P(b);
}
sx.close();
}
}
```



Output

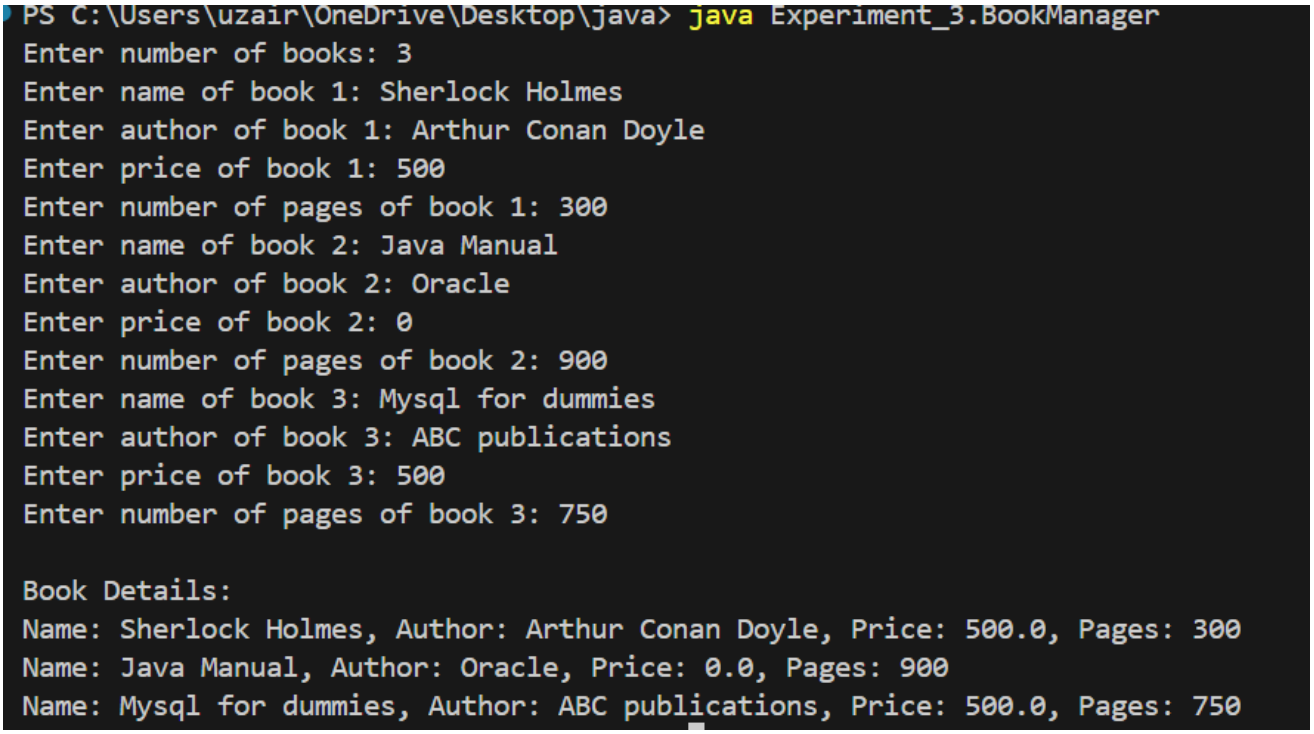
```
Enter number of books : 3
Enter name of book 1: Sherlock Holmes
Enter author of book 1 : Arthur Conan Doyle
Enter price of book 1 : 500
Enter number of pages of book 1 : 300
Enter name of book 2 : Java Manual
Enter author of book 2: Oracle
Enter price of book 2 : 0
Enter number of pages of book 2: 900
Enter name of book 3: ABC publications
Enter price of book 3 : 500
Enter number of pages of book 3: 750

Book Details:
Name : Sherlock Holmes, Author: Arthur Conan Doyle,
Price : 500.0, Pages: 300
Name: Java Manual, Author: Oracle, Price:0.0,
Pages : 900
Name : MySql for dummies, Author: ABC publications,
Price: 500.0, Pages: 750
```

24/10/24

Code:

```
package Experiment_3;

import java.util.Scanner;

class Book {
    private String name;
    private String author;
    private double price;
    private int numPages;

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

    static void setd(Book[] books) {
```

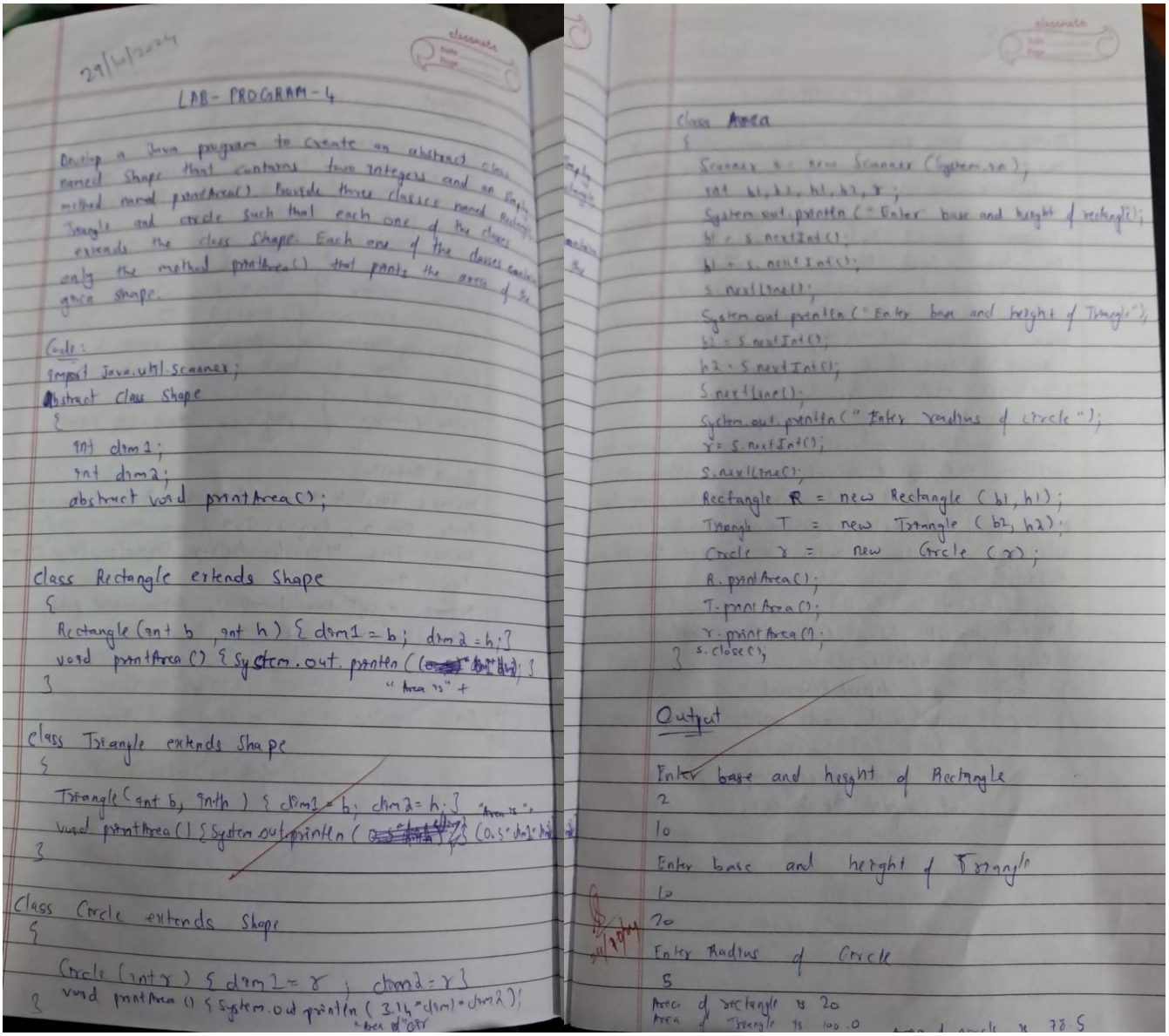
```
        Scanner sx = new Scanner(System.in);
        int n = books.length;
        for (int i = 0; i < n; i++) {
            System.out.print("Enter name of book " + (i + 1) + ": ");
            String name = sx.nextLine();
            System.out.print("Enter author of book " + (i + 1) + ": ");
            String author = sx.nextLine();
            System.out.print("Enter price of book " + (i + 1) + ": ");
            double price = sx.nextDouble();
            System.out.print("Enter number of pages of book " + (i + 1) + ": ");
            int numPages = sx.nextInt();
            sx.nextLine(); // Consume newline
            books[i] = new Book(name, author, price, numPages);
        }
    }

    String getd() {
        return "Name: " + name + ", Author: " + author + ", Price: " + price + ", Pages: " + numPages;
    }

    public String toString() {
        return getd();
    }
}

class BookManager {
    public static void main(String[] args) {
        Scanner sx = new Scanner(System.in);
        System.out.print("Enter number of books: ");
        int n = sx.nextInt();
        sx.nextLine(); // Consume newline
        Book[] books = new Book[n];

        Book.setd(books);

        System.out.println("\nBook Details:");
        for (Book b : books) {
            System.out.println(b);
        }

        sx.close();

    }
}
```


PS C:\Users\uzair\OneDrive\Desktop\java> java Experiment_3.BookManager
Enter number of books: 3
Enter name of book 1: Sherlock Holmes
Enter author of book 1: Arthur Conan Doyle
Enter price of book 1: 500
Enter number of pages of book 1: 300
Enter name of book 2: Java Manual
Enter author of book 2: Oracle
Enter price of book 2: 0
Enter number of pages of book 2: 900
Enter name of book 3: Mysql for dummies
Enter author of book 3: ABC publications
Enter price of book 3: 500
Enter number of pages of book 3: 750
Book Details:
Name: Sherlock Holmes, Author: Arthur Conan Doyle, Price: 500.0, Pages: 300
Name: Java Manual, Author: Oracle, Price: 0.0, Pages: 900
Name: Mysql for dummies, Author: ABC publications, Price: 500.0, Pages: 750

Output

Program 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 class Shape. Each one of the classes contain only the method printArea( ) that prints the area of the given shape.




```
abstract class Shape{
   int dim1;
   int dim2;
   abstract void printArea();
}

class Rectangle extends Shape{
   Rectangle(int b,int h){dim1=b;dim2=h;}
   void printArea(){System.out.println("Area of rectangle is "+dim1*dim2);}
}
class Triangle extends Shape{
   Triangle(int b,int h){dim1=b;dim2=h;}
   void printArea(){System.out.println("Area of triangle is "+0.5*dim1*dim2);}
}
```

```
class Circle extends Shape{
   Circle(int r){dim1=r;dim2=r;}
   void printArea(){System.out.println("Area of circle is "+3.14*dim1*dim2);}
}

class Area {
   public static void main(String[] args) {
      Scanner s = new Scanner(System.in);
      int b1,b2,h1,h2,r;
      System.out.println("Enter base and height of Rectangle ");
      b1 = s.nextInt();
      h1 = s.nextInt();
      s.nextLine();
      System.out.println("Enter base and height of Triangle ");
      b2 = s.nextInt();
      h2 = s.nextInt();
      s.nextLine();
      System.out.println("Enter Radius of Circle ");
      r = s.nextInt();
      s.nextLine();
      Rectangle R = new Rectangle(b1, h1);
      Triangle T = new Triangle(b2,h2);
      Circle C = new Circle(r);
      R.printArea();
      T.printArea();
      C.printArea();
      s.close();
   }
}
```

```
Enter base and height of Rectangle
2
10
Enter base and height of Triangle
10
20
Enter Radius of Circle
5
Area of rectangle is 20
Area of triangle is 100.0
Area of circle is 78.5
PS C:\Users\uzair\OneDrive\Desktop\java>
```

Output

Program 5

Develop a Java program to create a class Bank thatmaintains 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:

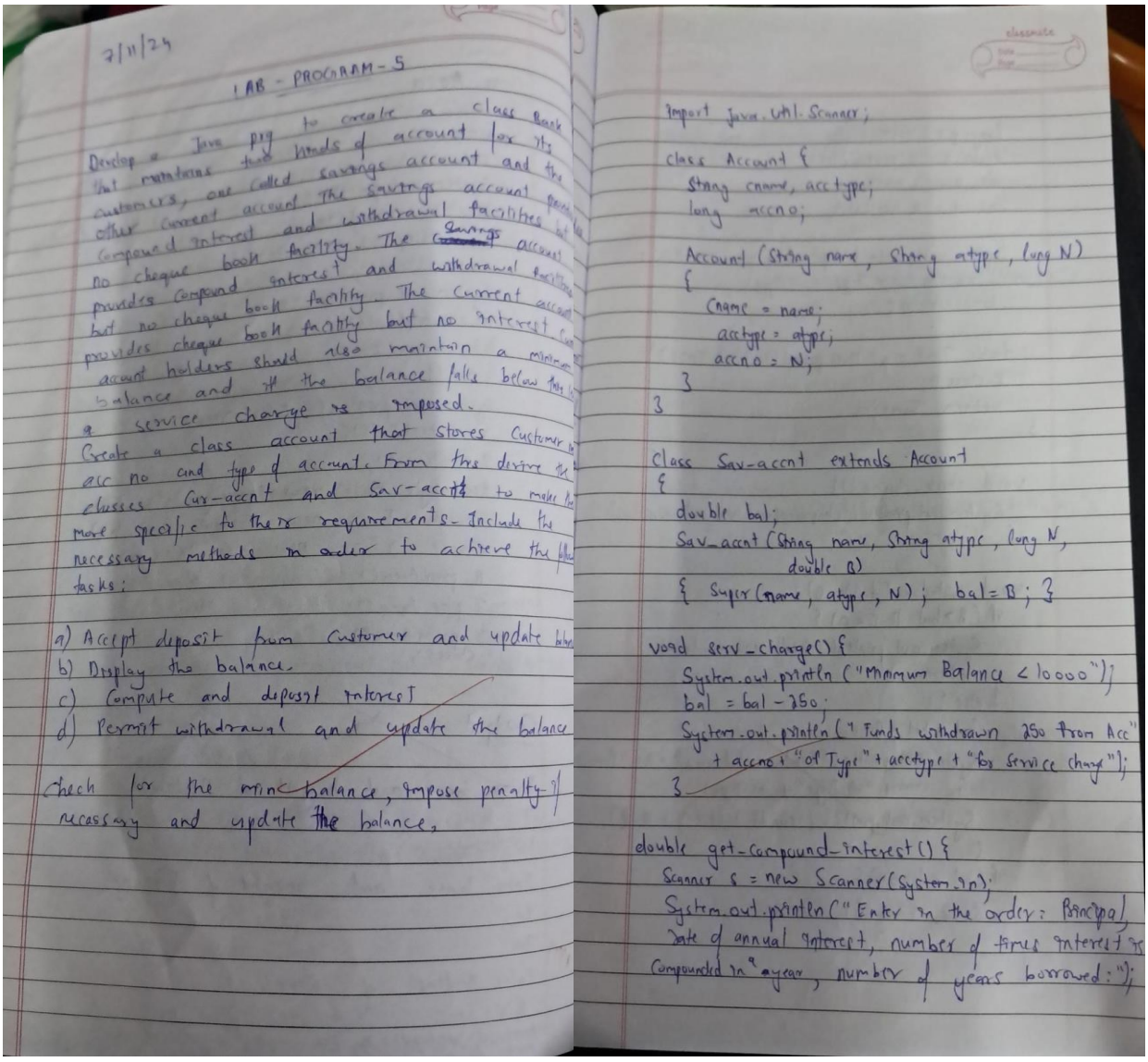
a) Accept deposit from customer and update the balance.

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.



7/11/24

LAB - PROGRAM - 5

Develop a Java pgm 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 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 a service charge is imposed.

Create a class account that stores Customer name, acc no 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:

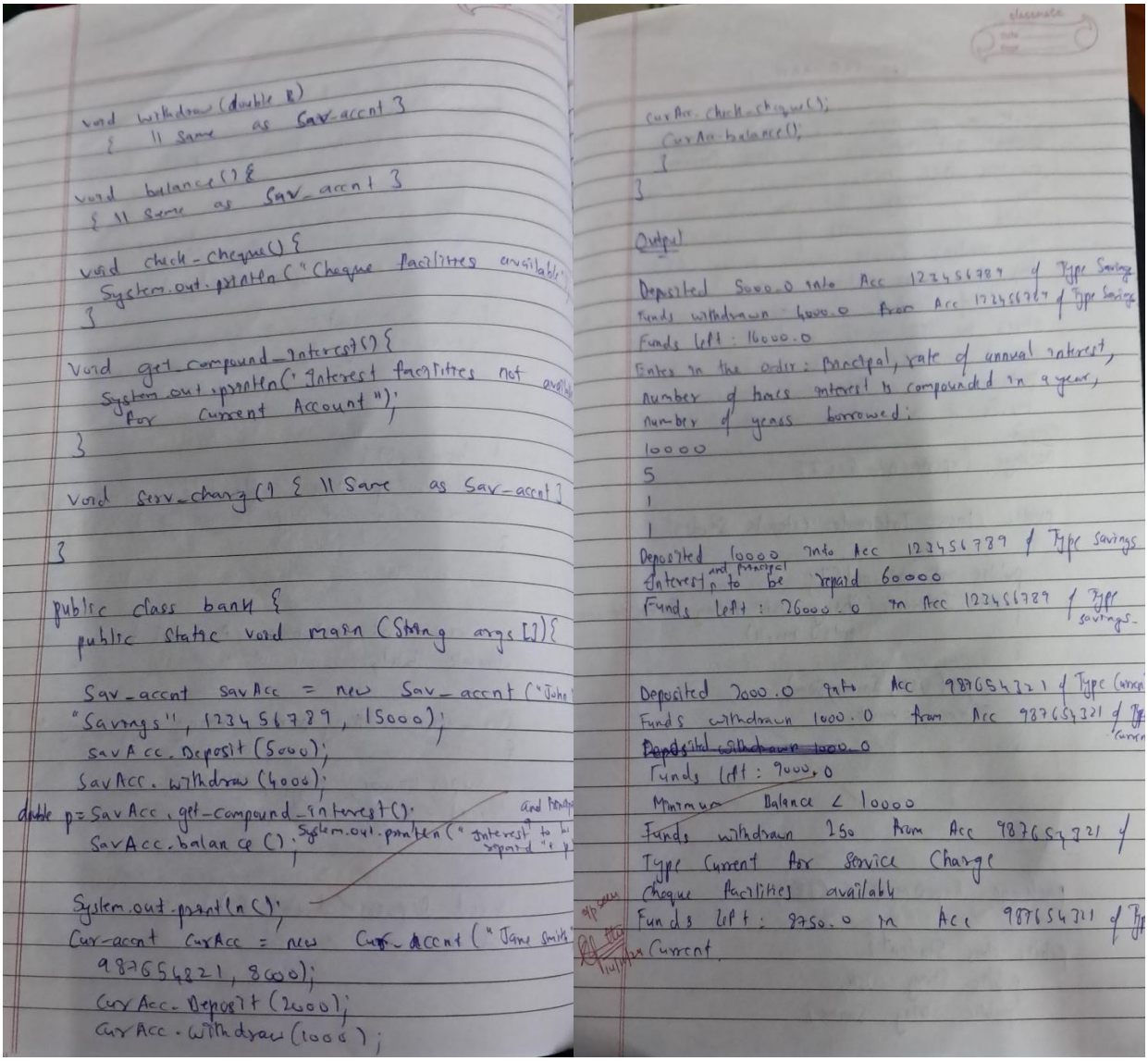
a) Accept deposit from customer and update balance
b) Display the balance.
c) Compute and deposit interest
d) Permit withdrawal and update the balance

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

```
import java.util.Scanner;

class Account {
    String cname, acctype;
    long accno;

    Account (String name, String atype, long N)
    {
      cname = name;
      acctype = atype;
      accno = N;
    }
}

class Sav-accnt extends Account
{
    double bal;
    Sav_accnt (String name, String atype, long N,
                double B)
    { super(name, atype, N); bal = B; }

void serv-charge() {
    System.out.println ("Minimum Balance < 10000");
    bal = bal - 250;
    System.out.println (" Funds withdrawn 250 from Acc"
      + accno + "of Type" + acctype + "for service charge");
}

double get-compound-interest() {
    Scanner s = new Scanner(System.in);
    System.out.println("Enter in the order: Principal
    rate of annual interest, number of times interest is
    Compounded in a year, number of years borrowed:");
```

```
long p = S.nextLong();
double r = S.nextDouble();
int n = S.nextInt();
int t = S.nextInt();

System.out.println("Deposited" + p + "into Acc" +
    accno + "of Type" + acctype);
bal = bal + p;

double Tamt = p * Math.pow((1+r/n), (n*t));
S.close();
return Tamt;
}

Void check_cheque(){
  System.out.println("cheque facilities not available
}

Void withdraw(double B){
  if(B>bal || B<0){
    System.out.println("Insufficient Funds");
    return;
  }else{
    bal = bal - B
    System.Out.println("Funds withdrawn" + B + "fro
    + accno + "of Type" + acctype);
    System.out.println("Funds left: " + bal);
  }
  if(bal < 10000){
    Serv_charge();
  }
}
```

```
Void Deposit(double B){
  if(B<0){
    System.out.println("Invalid Deposit Amount");
    return;
  }else{
    bal = bal + B;
    System.out.println("Deposited" + B + "into Acc" +
    accno + "of Type" + acctype);
  }
  if(bal<10000){
    serv_charge();
  }
}

Void balance(){
  System.out.println("Funds left: " + bal + "in Acc" +
    accno + "of Type" + acctype);
  }
}

Class Cur_accnt extends Account{
  double bal;

  Cur_accnt(String name, String atype, long N, double B)
  {
    super(name, atype, N);
    bal = B;
  }

  Void Deposit(double B){
  //Same as above
  }
```



Code:

```
import java.util.Scanner;

class Account {
    String cname, acctype;
    long accno;


    Account(String name, String atype, long N) {
        cname = name;
        acctype = atype;
        accno = N;
    }
}

class Sav_accnt extends Account {
    double bal;

    Sav_accnt(String name, String atype, long N, double B) {
        super(name, atype, N);
        bal = B;
    }

    void serv_charge() {
        System.out.println("Minimum Balance < 10000");
        bal = bal - 250;
```

```
        System.out.println("Funds withdrawn 250 from Acc " + accno + " of Type " + acctype + " for Service
Charge");
    }

    double get_compound_interest() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter in the order: Principal, rate of annual interest, number of times interest is
compounded in a year, number of years borrowed:");

        long p = s.nextLong();
        double r = s.nextDouble();
        int n = s.nextInt();
        int t = s.nextInt();

        System.out.println("Deposited " + p + " into Acc " + accno + " of Type " + acctype);
        bal = bal + p;

        double Tamt = p * Math.pow((1 + r / n), (n * t)); // Compound interest formula
        s.close();

        return Tamt;
    }

    void check_cheque() {
        System.out.println("Cheque facilities not available");
    }


    void withdraw(double B) {
        if (B > bal || B < 0) {
            System.out.println("Insufficient Funds");
            return;
        } else {
            bal = bal - B;
            System.out.println("Funds withdrawn " + B + " from Acc " + accno + " of Type " + acctype);
            System.out.println("Funds Left: " + bal);
        }
        if (bal < 10000) {
            serv_charge(); // Apply service charge if balance is less than 10,000
        }
    }

    void Deposit(double B) {
        if (B < 0) {
            System.out.println("Invalid Deposit Amount");
            return;
        } else {
            bal = bal + B;
            System.out.println("Deposited " + B + " into Acc " + accno + " of Type " + acctype);
        }
        if (bal < 10000) {
            serv_charge(); // Apply service charge if balance is less than 10,000
        }
    }
    void balance() {
        System.out.println("Funds Left: " + bal + " in Acc " + accno + " of Type " + acctype);
    }
}
```

```
class Cur_accnt extends Account {
    double bal;


    Cur_accnt(String name, String atype, long N, double B) {
        super(name, atype, N);
        bal = B;
    }


    void Deposit(double B) {
        if (B < 0) {
            System.out.println("Invalid Deposit Amount");
            return;
        } else {
            bal = bal + B;
            System.out.println("Deposited " + B + " into Acc " + accno + " of Type " + acctype);
        }
        if (bal < 10000) {
            serv_charge();
        }
    }

    void withdraw(double B) {
        if (B > bal || B < 0) {
            System.out.println("Insufficient Funds");
            return;
        } else {
            bal = bal - B;
            System.out.println("Funds withdrawn " + B + " from Acc " + accno + " of Type " + acctype);
            System.out.println("Funds Left: " + bal);
        }
        if (bal < 10000) {
            serv_charge();
        }
    }


    void balance() {
        System.out.println("Funds Left: " + bal + " in Acc " + accno + " of Type " + acctype);
    }

    void check_cheque() {
        System.out.println("Cheque facilities available");
    }


    void get_compound_interest() {
        System.out.println("Interest facilities not available for Current Account");
    }


    void serv_charge() {
        System.out.println("Minimum Balance < 10000");
        bal = bal - 250;
        System.out.println("Funds withdrawn 250 from Acc " + accno + " of Type " + acctype + " for Service
Charge");
```

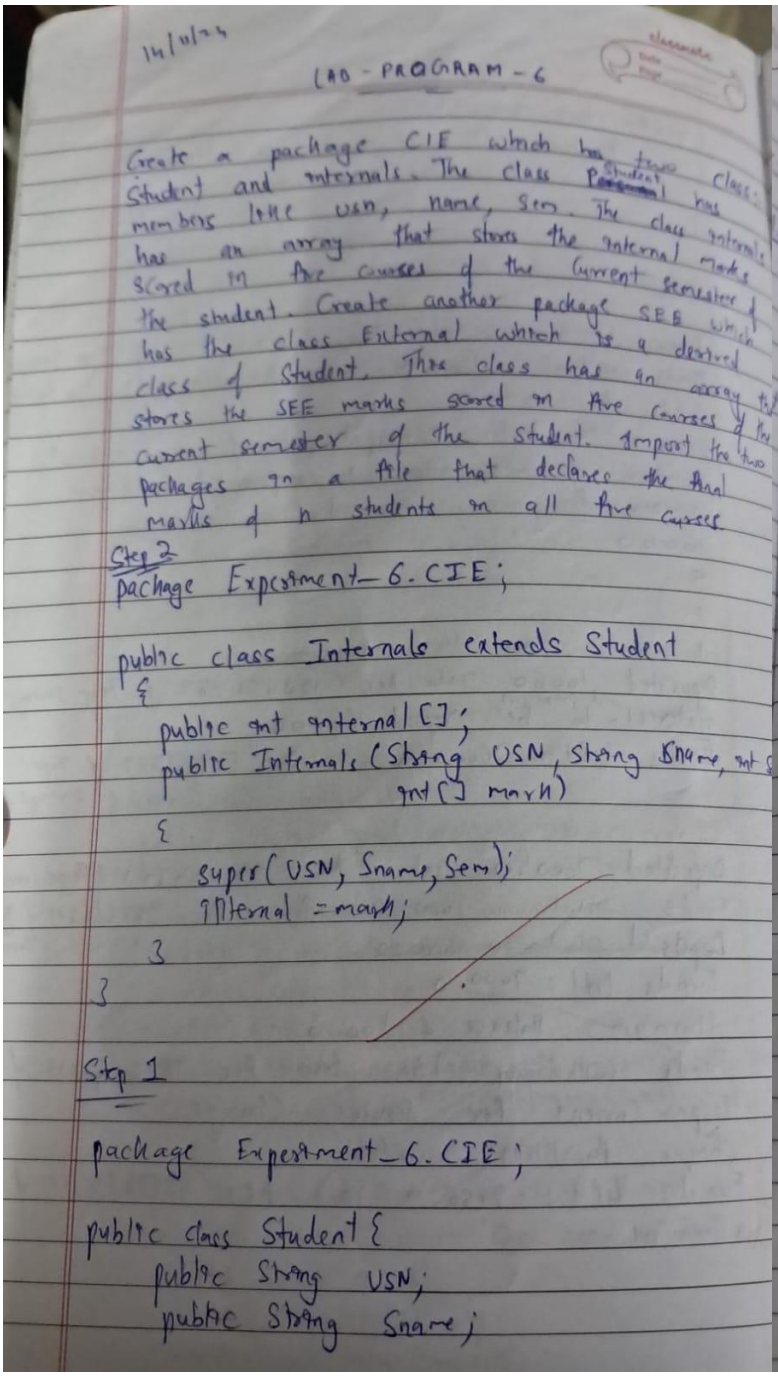
```
    }
}

public class bank {
    public static void main(String[] args) {
        Sav_accnt savAcc = new Sav_accnt("John Doe", "Savings", 123456789, 15000);
        savAcc.Deposit(5000);
        savAcc.withdraw(4000);
        savAcc.get_compound_interest();
        savAcc.balance();

        System.out.println("\n BELOW STARTS CURRENT ACCOUNT TRANSACTIONS ");

        Cur_accnt curAcc = new Cur_accnt("Jane Smith", "Current", 987654321, 8000);
        curAcc.Deposit(2000);
        curAcc.withdraw(1000);
        curAcc.check_cheque();
        curAcc.balance();
    }
}
```

```
C:\Users\uzair\OneDrive\Desktop\java\Experiment_5>java bank
Deposited 5000.0 into Acc 123456789 of Type Savings
Funds withdrawn 4000.0 from Acc 123456789 of Type Savings
Funds Left: 16000.0
Enter in the order: Principal, rate of annual interest, number of times interest is compounded in a year, number of year
s borrowed:
10000
2
3
2
Deposited 10000 into Acc 123456789 of Type Savings
Funds Left: 26000.0 in Acc 123456789 of Type Savings

 BELOW STARTS CURRENT ACCOUNT TRANSACTIONS
Deposited 2000.0 into Acc 987654321 of Type Current
Funds withdrawn 1000.0 from Acc 987654321 of Type Current
Funds Left: 9000.0
Minimum Balance < 10000
Funds withdrawn 250 from Acc 987654321 of Type Current for Service Charge
Cheque facilities available
Funds Left: 8750.0 in Acc 987654321 of Type Current
```

Output

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.



14/11/24

LAB - PROGRAM - 6

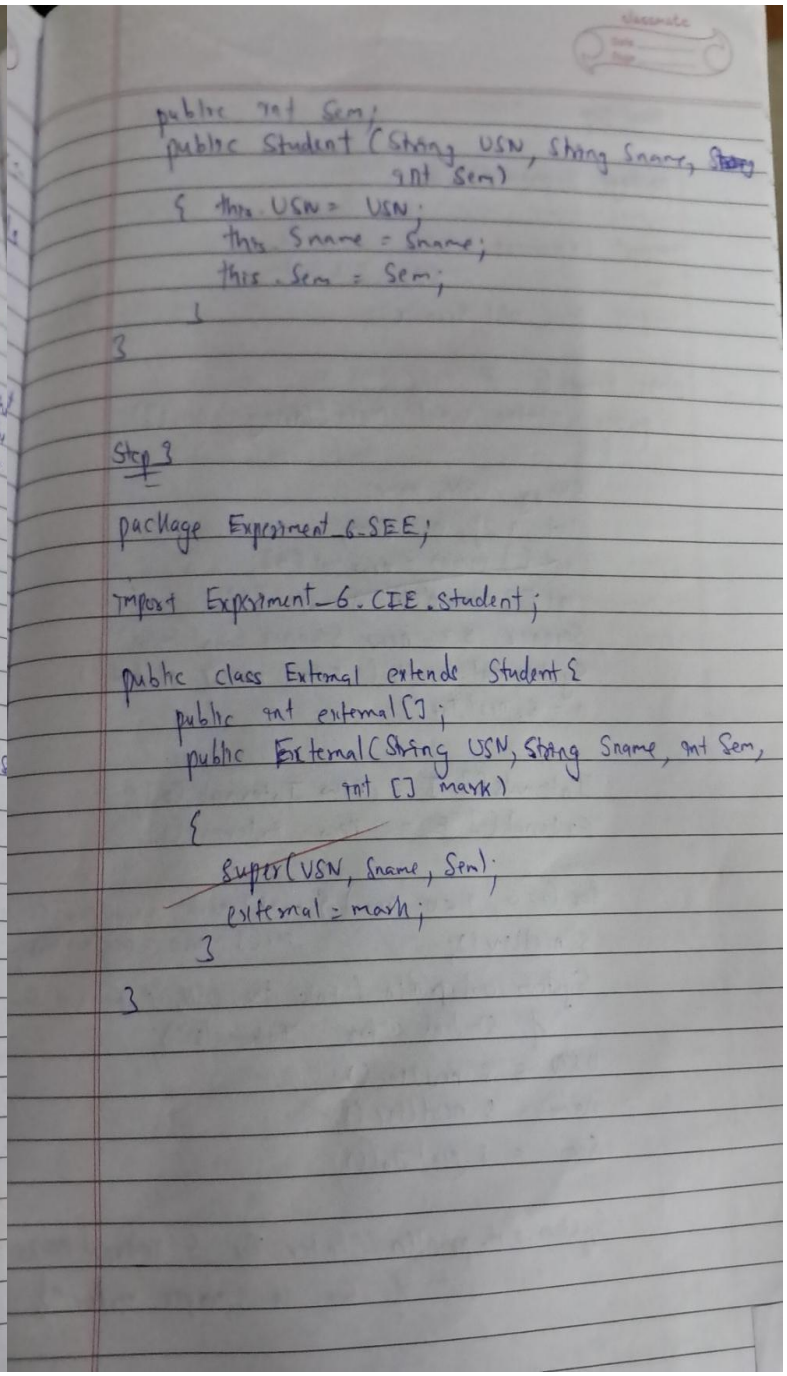
Create a package CIE which has two classes Student and Internals. The class Personal has members like usn, name, sem. The class internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External 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.

Step 2

```
package Experiment_6.CIE;

public class Internals extends Student
{
    public int internal[];
    public Internals(String USN, String Sname, int
                     int[] mark)
    {
        super(USN, Sname, Sem);
        Internal = mark;
    }
}
```

Step 1

```
package Experiment_6.CIE;

public class Student{
    public String USN;
    public String Sname;
```



```
    public int Sem;
    public Student(String USN, String Sname,
                   int Sem)
    { this.USN = USN;
      this.Sname = Sname;
      this.Sem = Sem;
    }
}
```

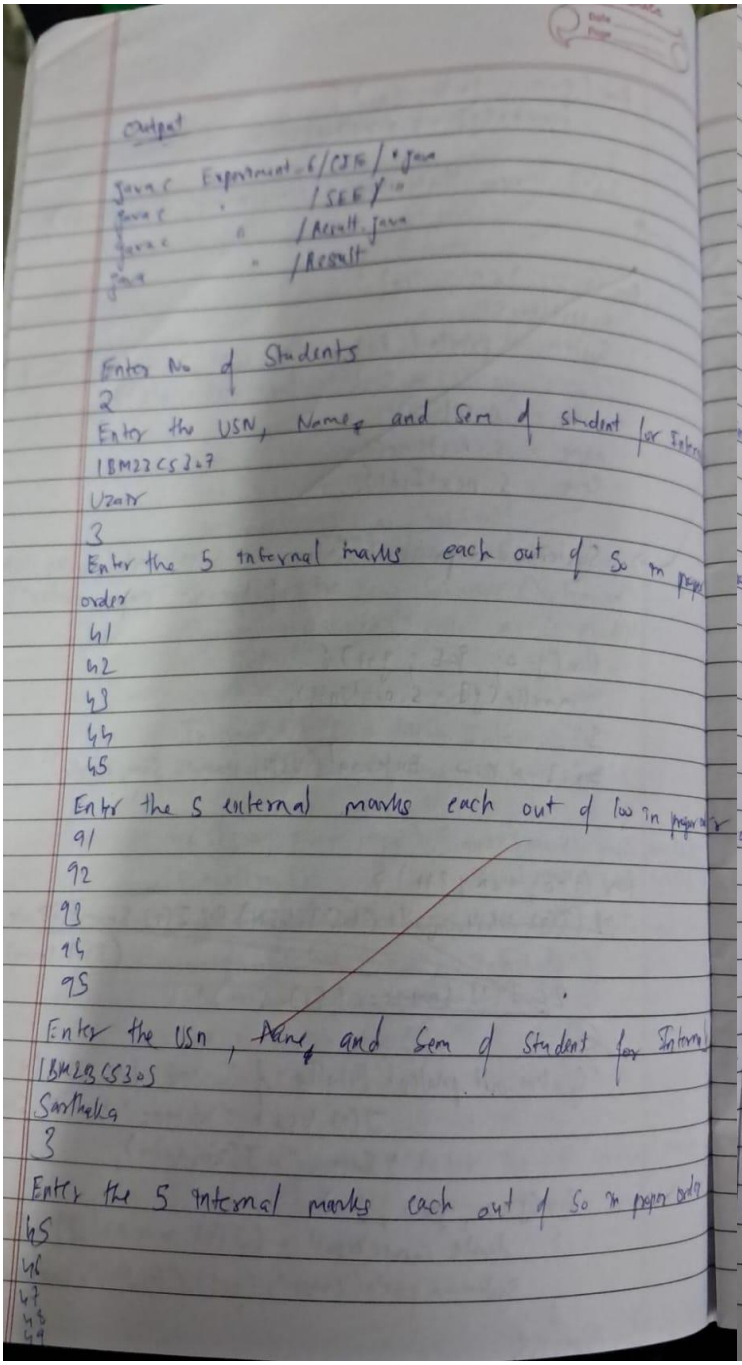
Step 3

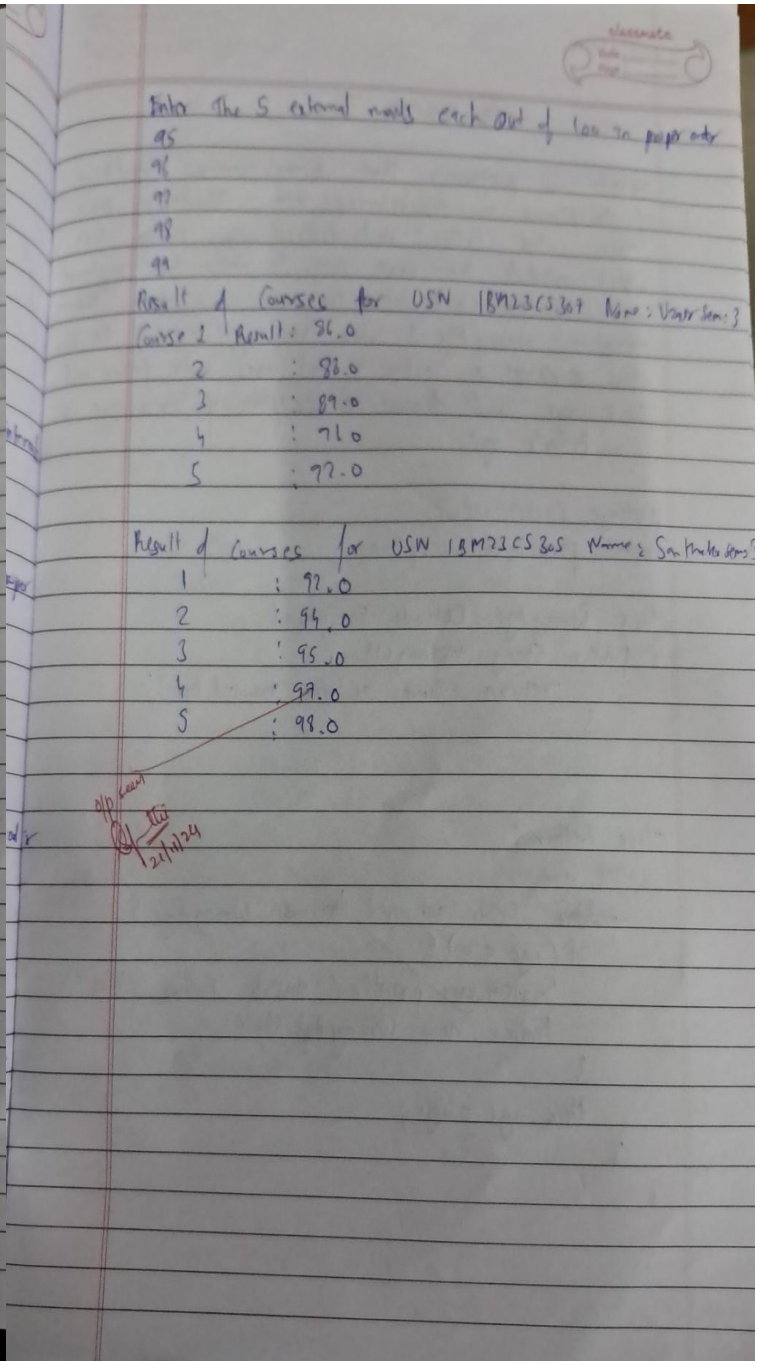
```
package Experiment_6.SEE;

import Experiment_6.CIE.Student;

public class External extends Student{
    public int external[];
    public External(String USN, String Sname, int Sem,
                    int [] mark)
    {
        super(USN, Sname, Sem);
        external = mark;
    }
}
```

Final Step

```
package Experiment_6;

import Experiment_6.CIE.Internals;
       "        .SEE.External;
import java.util.Scanner;

class Result {
  public static void main(String args[])
  {
    String USN, name;
    int i, j, n, Sem;
    int[] marks = new int[5];
    int[] marks = new int[5];
    Scanner s = new Scanner(System.in);
    System.out.println("Enter no of students");
    n = s.nextInt();

    Internals[] I = new Internals[n];
    External[] E = new External[n];

    for(i=0; i<n; i++) { int[] marks = new int[5];
    s.nextLine();                int[] marks = new int[5];
    System.out.println("Enter the USN, Name and Sem
                        of student for Internals");
    USN = s.nextLine();
    name = s.nextLine();
    Sem = s.nextInt();

    System.out.println("Enter The 5 internal marks
        each out of 50 in proper order");
```

```
    for(j=0; j<5; j++) {
      marks[j] = s.nextInt();
    }
    I[i] = new Internals(USN, name, Sem, marks);

    for(i=0; i<n; i++) {
      s.nextLine();
      System.out.println("Enter the USN, Name, and Sem of
                          student for externals");
      USN = s.nextLine();
      name = s.nextLine();
      Sem = s.nextInt();

      System.out.println("Enter the 5 external marks each
                          out of 100 in paper order");

      for(j=0; j<5; j++) {
        marks[j] = s.nextInt();
      }
      E[i] = new External(USN, name, Sem, marks);
    }

    for(i=0; i<n; i++) {
      if(I[i].USN.equals(E[i].USN) && I[i].Sname.equals
                                      (E[i].Sname)
        && I[i].Sem == E[i].Sem)

      System.out.println("Results of courses for USN "+
          I[i].USN + "Name: " + I[i].Sname +
          "Sem: " + I[i].Sem);
      for(j=0; j<5; j++)
      { double CourseResult = (I[i].internal[j] + E[i].external
        System.out.println("Course "+(j+1)+"Result: "+CourseResult);
      }
    } s.close();}
```





Code:

```
package Experiment_6.CIE;

public class Student{
    public String USN;
    public String Sname;
    public int Sem;
    public Student(String USN,String Sname,int Sem){
        this.USN=USN;
        this.Sname=Sname;
        this.Sem=Sem;
    }
}
```

```
package Experiment_6.SEE;

import Experiment_6.CIE.Student;

public class External extends Student{
    public int external[];
    public External(String USN,String Sname,int Sem,int [] mark){
    super(USN,Sname,Sem);
    external=mark;
    }
}


package Experiment_6;

import Experiment_6.CIE.Internals;
import Experiment_6.SEE.External;
import java.util.Scanner;

class Result {
    public static void main(String args[]) {
        String Usn, name;
        int i, j, n, Sem;

        Scanner s = new Scanner(System.in);

        System.out.println("Enter No of Students");
        n = s.nextInt();

        Internals[] I = new Internals[n];
        External[] E = new External[n];


        for(i = 0; i < n; i++) {
            int[] marki = new int[5];
            int[] marke = new int[5];
            s.nextLine();
            System.out.println("Enter The Usn, Name and Sem of student For Internals");
            Usn = s.nextLine();
            name = s.nextLine();
            Sem = s.nextInt();

            System.out.println("Enter The 5 internal marks each out of 50 in proper order");
            for (j = 0; j < 5; j++) {
                marki[j] = s.nextInt();
            }
            I[i] = new Internals(Usn, name, Sem, marki);

            System.out.println("Enter The 5 external marks each out of 100 in proper order");
            for (j = 0; j < 5; j++) {
                marke[j] = s.nextInt();
            }
            E[i] = new External(Usn, name, Sem, marke);
        }
```

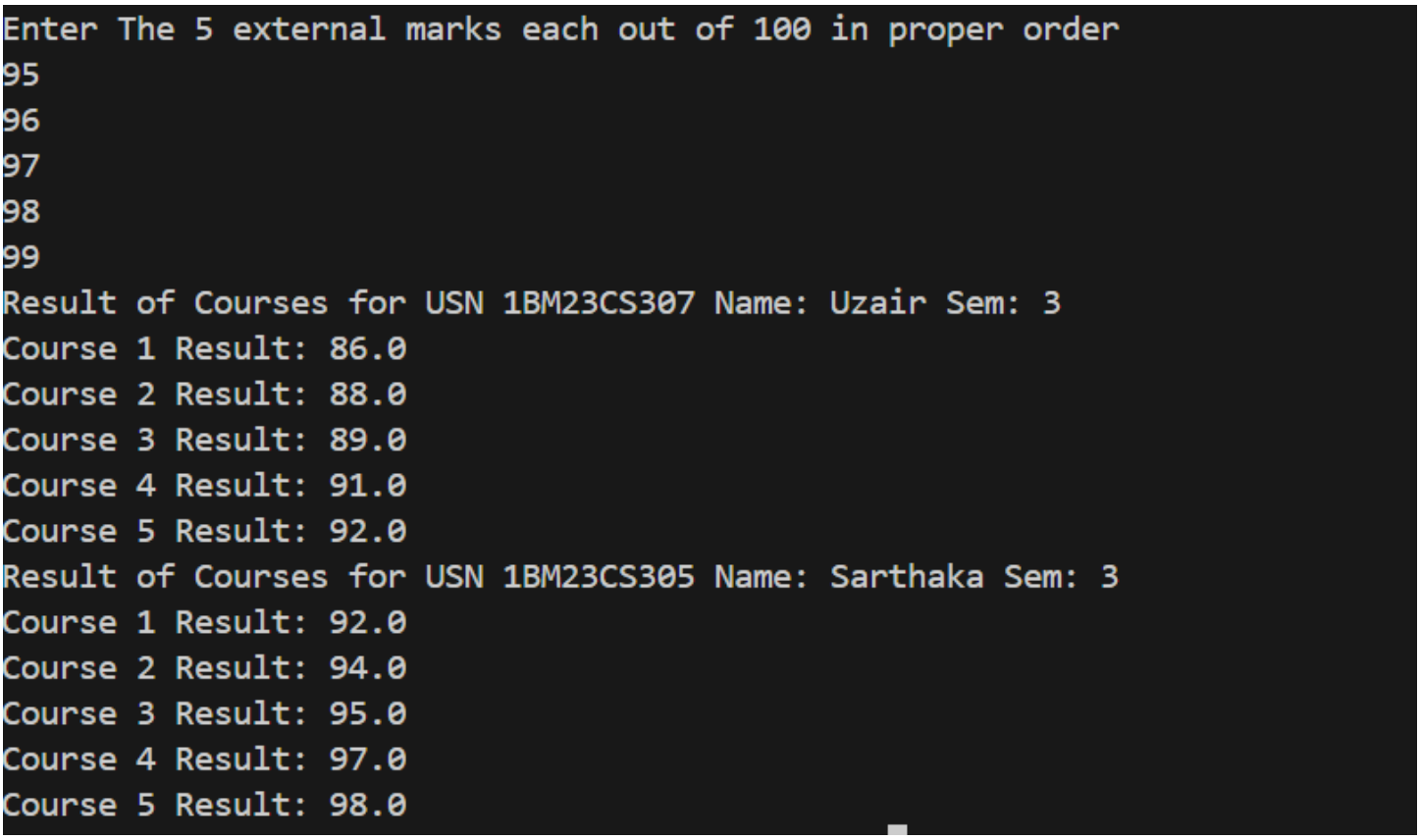
```
        // Print results
        for(i = 0; i < n; i++) {

            System.out.println("Result of Courses for USN " + I[i].USN + " Name: " + I[i].Sname + " Sem: "
+ I[i].Sem);
            for (j = 0; j < 5; j++) {
               double courseResult = ((I[i].internal[j] + ((E[i].external[j])/2)));
               System.out.println("Course " + (j + 1) + " Result: " + courseResult);
            }

        }

        s.close();
    }
}
```

```
PS C:\Users\uzair\OneDrive\Desktop\java> cd .\Java\
PS C:\Users\uzair\OneDrive\Desktop\java\Java> javac Experiment_6/CIE/*java
PS C:\Users\uzair\OneDrive\Desktop\java\Java> javac Experiment_6/SEE/*java
PS C:\Users\uzair\OneDrive\Desktop\java\Java> javac Experiment_6/Result.java
PS C:\Users\uzair\OneDrive\Desktop\java\Java> java Experiment_6/Result
```

Output 1

```
Enter No of Students
2
Enter The Usn, Name and Sem of student For Internals
1BM23CS307
Uzair
3
Enter The 5 internal marks each out of 50 in proper order
41
42
43
44
45
Enter The 5 external marks each out of 100 in proper order
91
92
93
94
95
Enter The Usn, Name and Sem of student For Internals
1BM23CS305
Sarthaka
3
Enter The 5 internal marks each out of 50 in proper order
45
46
47
48
49
Enter The 5 external marks each out of 100 in proper order
95
96
97
98
```

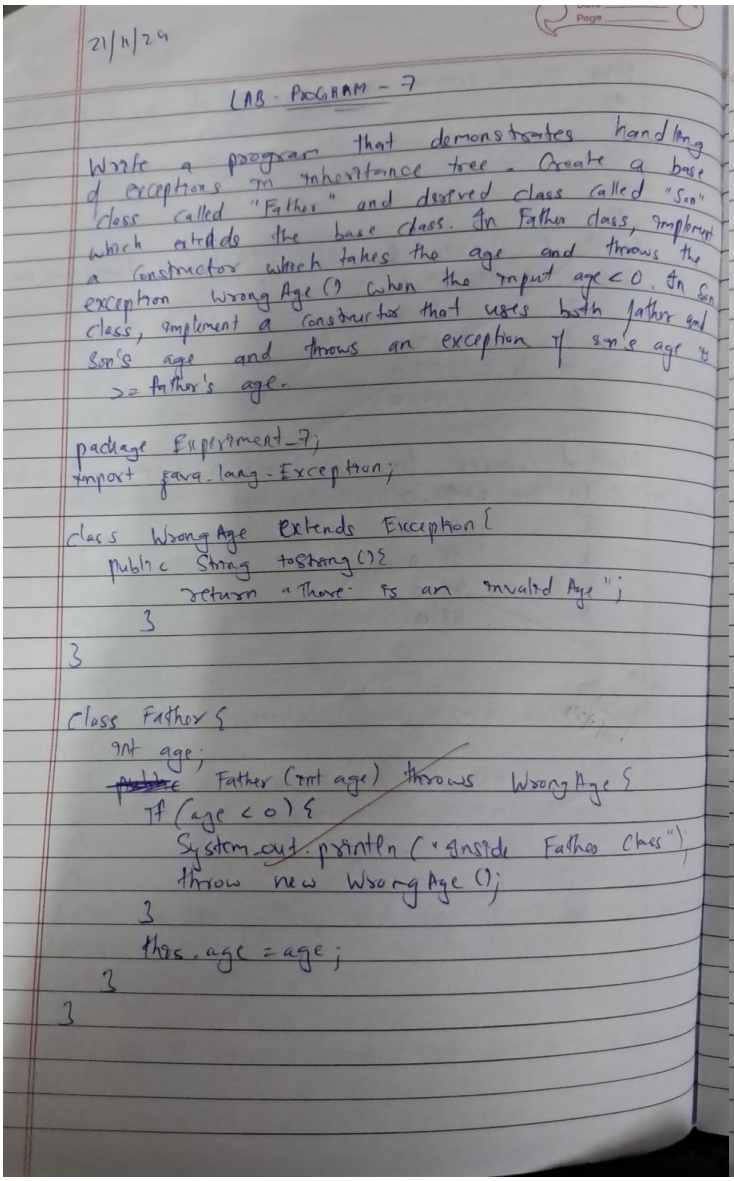
Output 2


Enter The 5 external marks each out of 100 in proper order
95
96
97
98
99
Result of Courses for USN 1BM23CS307 Name: Uzair Sem: 3
Course 1 Result: 86.0
Course 2 Result: 88.0
Course 3 Result: 89.0
Course 4 Result: 91.0
Course 5 Result: 92.0
Result of Courses for USN 1BM23CS305 Name: Sarthaka Sem: 3
Course 1 Result: 92.0
Course 2 Result: 94.0
Course 3 Result: 95.0
Course 4 Result: 97.0
Course 5 Result: 98.0

Output 3

Program 7

Write a program that demonstrates handling ofexceptions 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 uses both father and son's age and throws an exception if son's age is >=father's age.

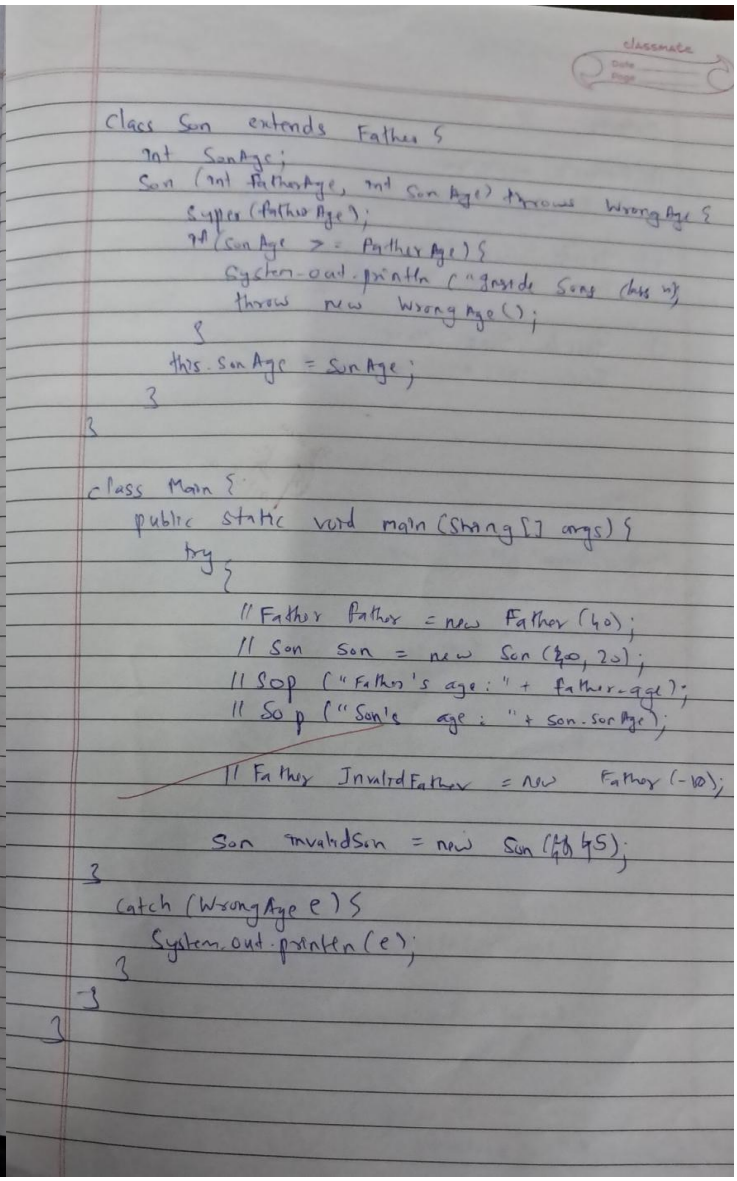


21/11/24

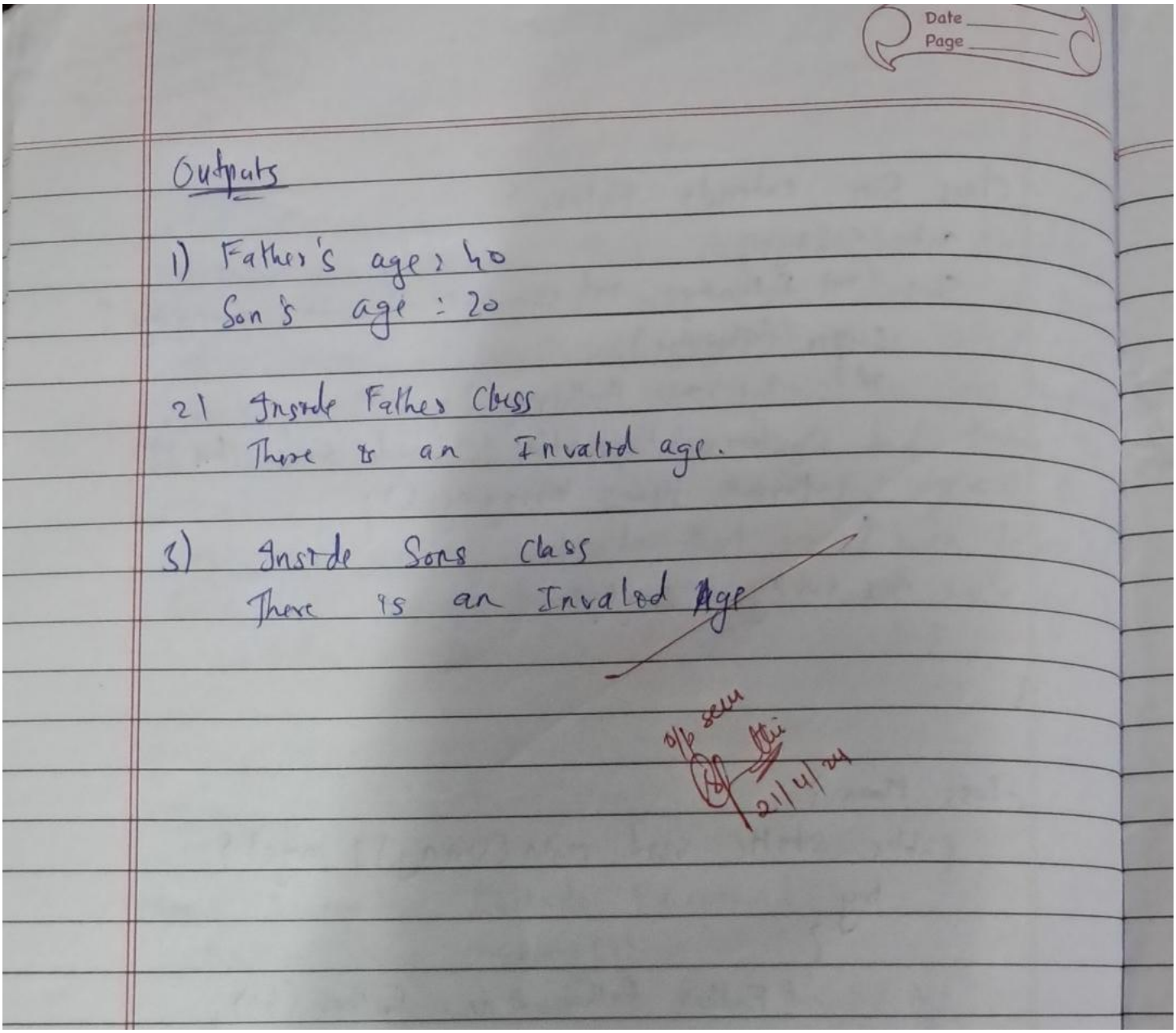
LAB PROGRAM - 7

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

```
package Experiment_7;
import java.lang.Exception;

class WrongAge extends Exception{
    public String toString(){
        return "There is an invalid Age";
    }
}

class Father{
    int age;
    Father(int age) throws WrongAge{
        if(age<0){
            System.out.println("Inside Father Class");
            throw new WrongAge();
        }
        this.age=age;
    }
}
```



```
class Son extends Father{
    int SonAge;
    Son(int FatherAge, int SonAge) throws WrongAge{
        super(FatherAge);
        if(SonAge >= FatherAge){
            System.out.println("Inside Son class");
            throw new WrongAge();
        }
        this.SonAge = SonAge;
    }
}

class Main{
    public static void main(String[] args){
        try{
            // Father father = new Father(40);
            // Son son = new Son(40,20);
            // Sop("Father's age:" + father.age);
            // Sop("Son's age:" + son.SonAge);

            // Father InvalidFather = new Father(-10);

            Son invalidSon = new Son(40,45);
        }
        catch(WrongAge e){
            System.out.println(e);
        }
    }
}
```



Code:

```
package Experiment_7;
import java.lang.Exception;

class WrongAge extends Exception{
    public String toString(){
        return "There is an invalid Age";
    }
}


class Father {
    int age;

    Father(int age) throws WrongAge {
        if (age < 0) {
```

```
            System.out.println("Inside Father Class");
            throw new WrongAge();
        }
        this.age = age;
    }
}


class Son extends Father {
    int sonAge;

    Son(int fatherAge, int sonAge) throws WrongAge {
        super(fatherAge);
        if (sonAge >= fatherAge) {
            System.out.println("Inside Sons Class");
            throw new WrongAge();
        }
        this.sonAge = sonAge;
    }
}

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

            //Father father = new Father(40);


            //Son son = new Son(40, 20);
            //System.out.println("Father's age: " + father.age);
            //System.out.println("Son's age: " + son.sonAge);


            //Father invalidFather = new Father(-10);


            Son invalidSon = new Son(40, 45);

        } catch (WrongAge e) {
            System.out.println(e);
        }
    }
}
```

```
PS C:\Users\uzair\OneDrive\Desktop\java\Java> java Experiment_7/Main
Father's age: 40
Son's age: 20
```

Output 1

```
PS C:\Users\uzair\OneDrive\Desktop\java\Java> java Experiment_7/Main
Inside Father Class
There is an invalid Age
```

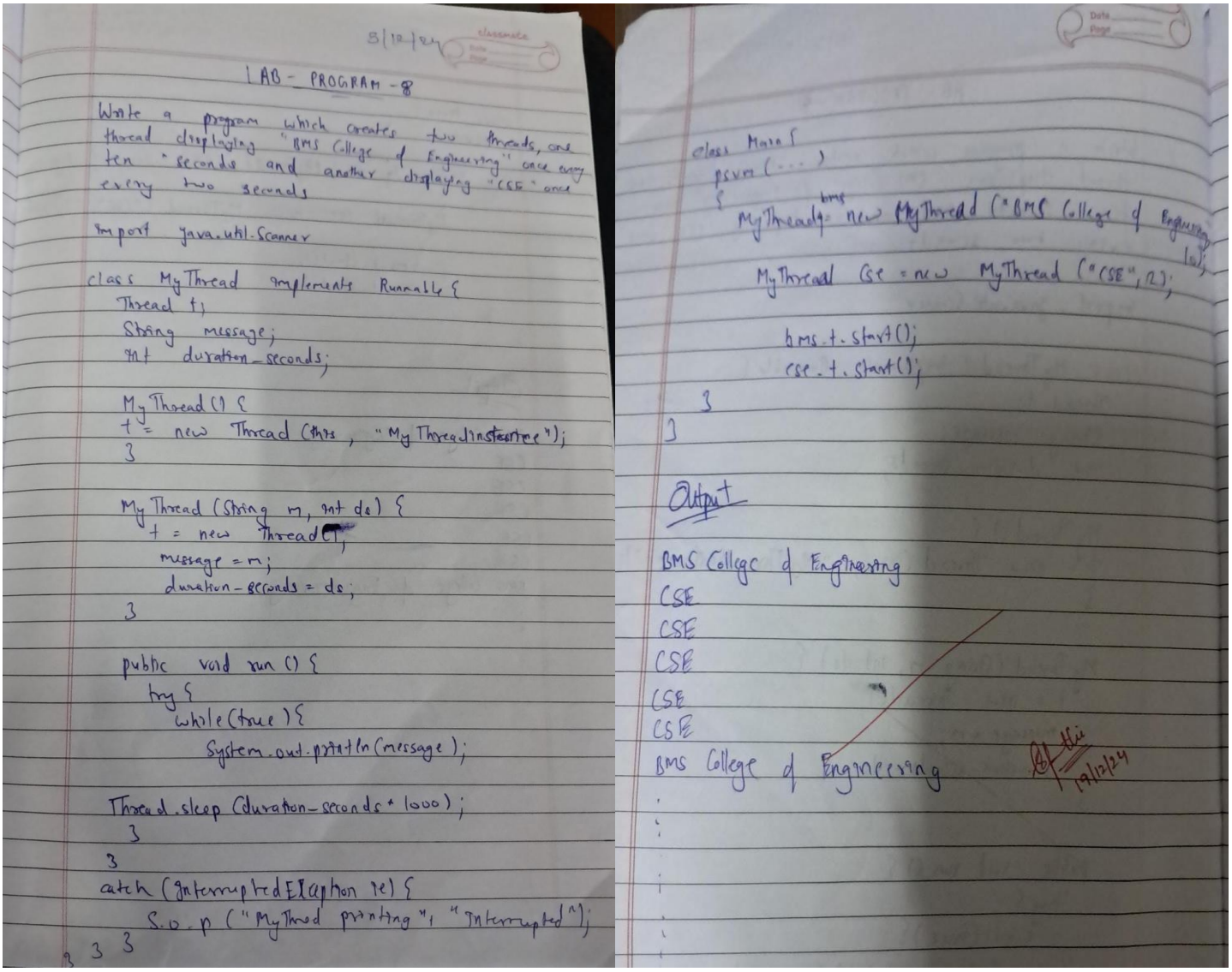
Output 2

```
PS C:\Users\uzair\OneDrive\Desktop\java\Java> java Experiment_7/Main
Inside Sons Class
There is an invalid Age
```

Output 3

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.



LAB - 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

```
import java.util.Scanner

class MyThread implements Runnable {
    Thread t;
    String message;
    int duration_seconds;

    MyThread() {
        t = new Thread(this, "MyThreadInstance");
    }

    MyThread(String m, int ds) {
        t = new Thread(this, "MyThreadInstance");
        message = m;
        duration_seconds = ds;
    }

    public void run() {
        try {
            while(true) {
                System.out.println(message);

                Thread.sleep(duration_seconds * 1000);
            }
        }
        catch (InterruptedException ie) {
            S.o.p("MyThread printing", "Interrupted");
        }
    }
}

class Main {
    psvm(...)
    {
        MyThread bms = new MyThread("BMS College of Engineering", 10);
        MyThread cse = new MyThread("CSE", 2);

        bms.t.start();
        cse.t.start();
    }
}
```

Output

```
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
.
.
.
.
.
```

14/12/24

Code:

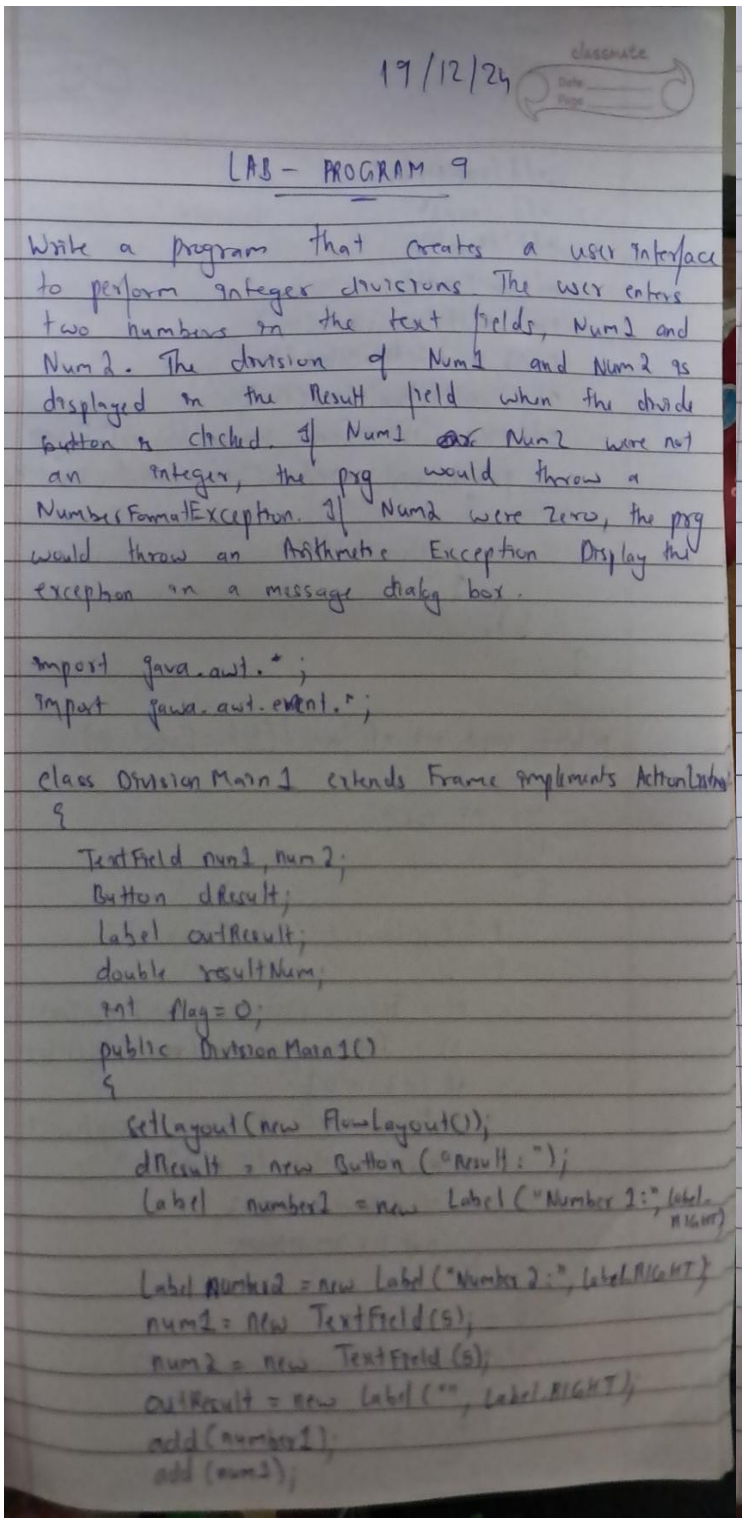
```
package Experiment_8;

class MyThread implements Runnable {
    Thread t;
    String message;
    int duration_seconds;

    MyThread() {
        t = new Thread(this, "MyThreadInstance");
    }

    MyThread(String m, int ds) {
        t = new Thread(this, "MyThreadInstance");
        message = m;
        duration_seconds = ds;
    }
```

```
    public void run() {
        try {
            while (true) {
                System.out.println(message);
                Thread.sleep(duration_seconds * 1000);
            }
        } catch (InterruptedException ie) {
            System.out.println("My Thread printing Interrupted");
        }
    }
}

class MainThread {
    public static void main(String[] args) {
        MyThread bms = new MyThread("BMS College Of Engineering", 10);
        MyThread cse = new MyThread("CSE", 2);
        bms.t.start();
        cse.t.start();
    }
}
```

Output

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.

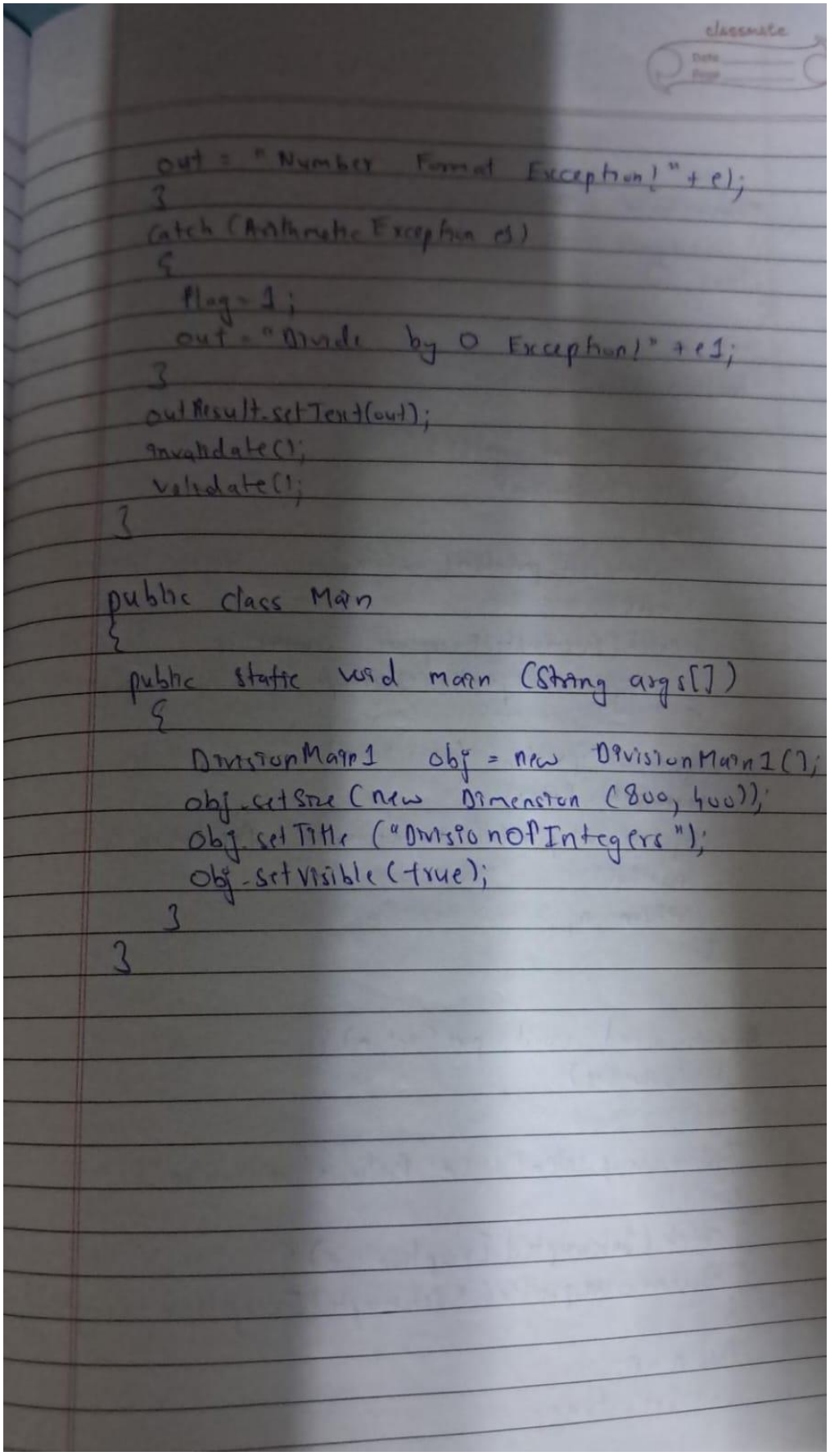


19/12/24

LAB - 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 prg would throw a NumberFormatException. If Num2 were zero, the prg would throw an Arithmetic Exception Display the exception in a message dialog box.

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

class DivisionMain1 extends Frame implements ActionListener
{
  TextField num1, num2;
  Button dResult;
  Label outResult;
  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("", Label.RIGHT);
    add(number1);
    add(num1);
    add(number2);
    add(num2);
    add(dResult);
    add(outResult);
    num1.addActionListner(this);
    num2.addActionListner(this);
    dResult.addActionListner(this);
    addWindowListner(new WindowAdapter() {
      public void windowClosing(WindowEvent e)
      {
        System.exit(0);
      }
    });
  }

  public void actionPerformed(ActionEvent e)
  {
    int n1, n2;
    try
    {
      if(e.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 += resultNum;
      }
    }
    catch(NumberFormatException e1)
    {
      flag = 1;
```



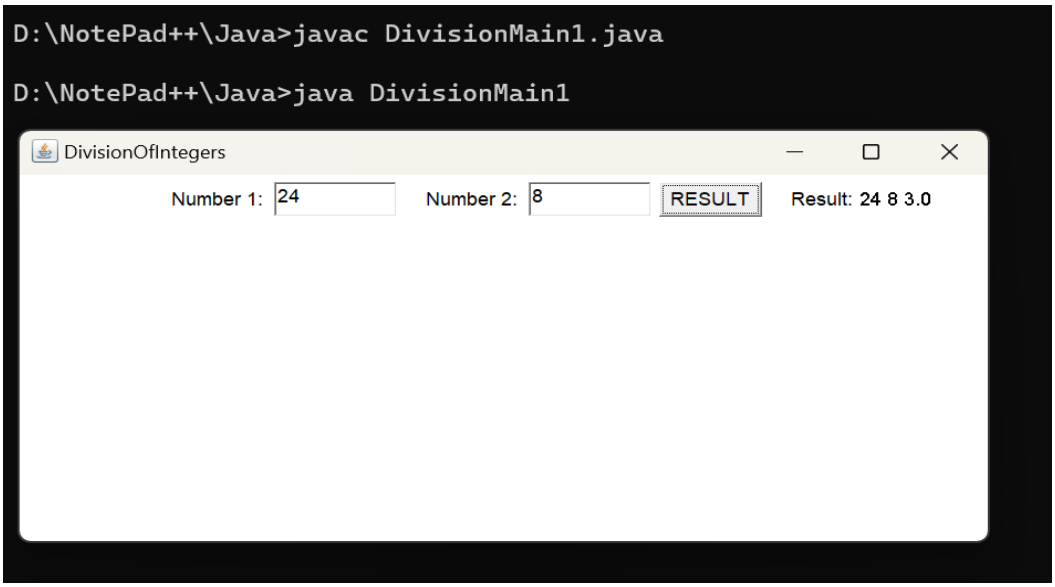
```
out = "Number Format Exception!" + e);
}
catch (ArithmeticException ob)
{
  flag=1;
  out = "Divide by 0 Exception!" + e1;
}
owlResult.setText(out);
invalidate();
validate();
}

public class Main
{
  public static void main (String args[])
  {
    DivisionMain1 obj = new DivisionMain1();
    obj.setSize(new Dimension(800, 400));
    obj.setTitle("Division of Integers");
    obj.setVisible(true);
  }
}
```

Code:

```
import java.awt.*;
import java.awt.event.*;
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("",Label.RIGHT);
        add(number1);
        add(num1);
        add(number2);
        add(num2);
        add(dResult);
        add(outResult);
        num1.addActionListener(this);
        num2.addActionListener(this);
        dResult.addActionListener(this);
        addWindowListener(new WindowAdapter(){
                public void windowClosing(WindowEvent e)
                {
                        System.exit(0);
                }
        });
        }
        public void actionPerformed(ActionEvent e)
        {
        int n1,n2;
        try
        {
                if (e.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+=resultNum;
                }
        }
        catch(NumberFormatException e1)
        {
                flag=1;
                out="Number Format Exception!"+e1;
        }
```

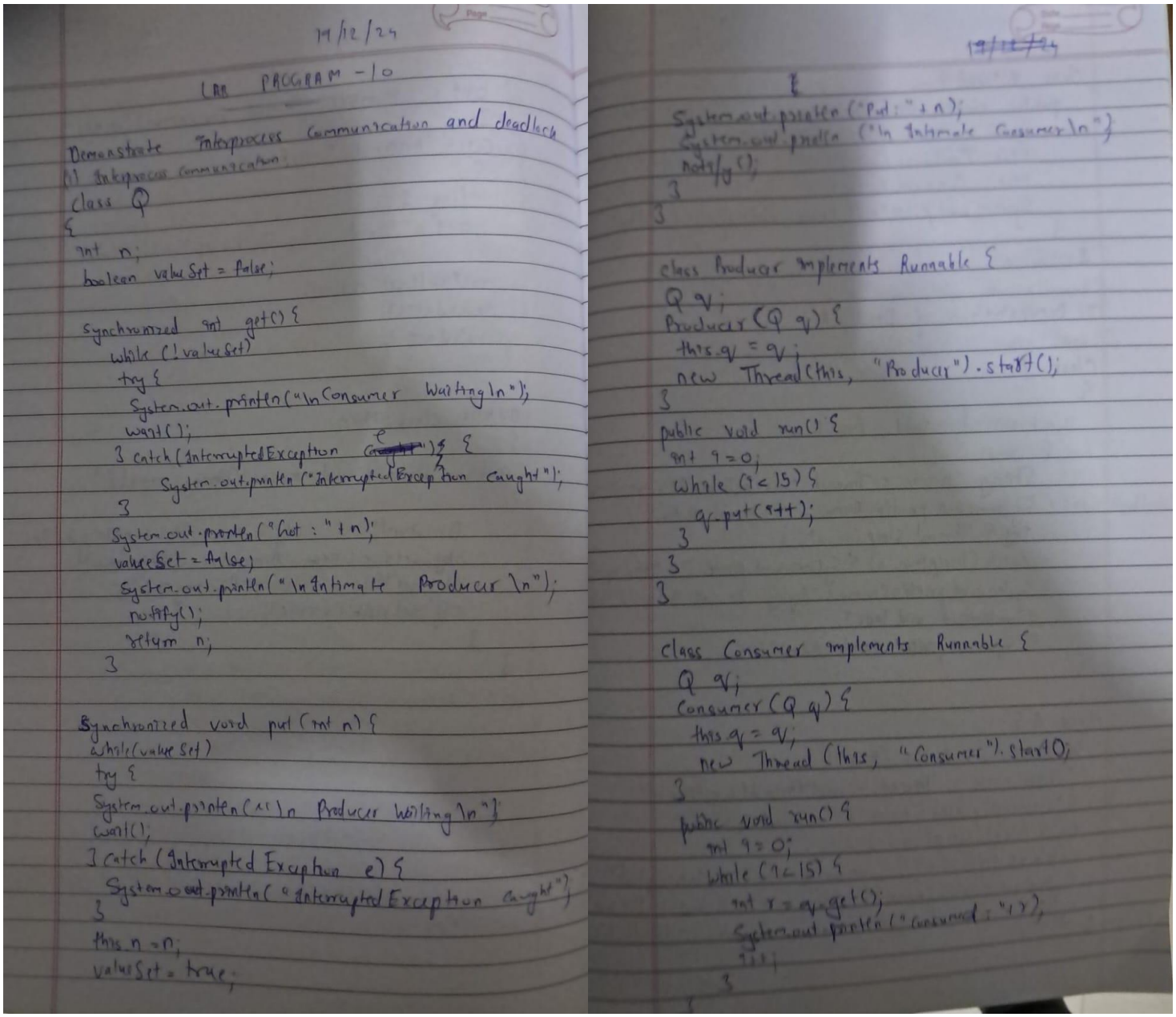
```
                    catch(ArithmeticException e1)
                    {
                            flag=1;
                            out="Divide by 0 Exception!"+e1;
                    }
                    outResult.setText(out);
                    invalidate();
                    validate();
                    }

}

public class Main
{
                    public static void main(String args[])
                    {
                    DivisionMain1 obj=new DivisionMain1();
                    obj.setSize(new Dimension(800,400));
                    obj.setTitle("DivisionOfIntegers");
                    obj.setVisible(true);
                    }
}
```



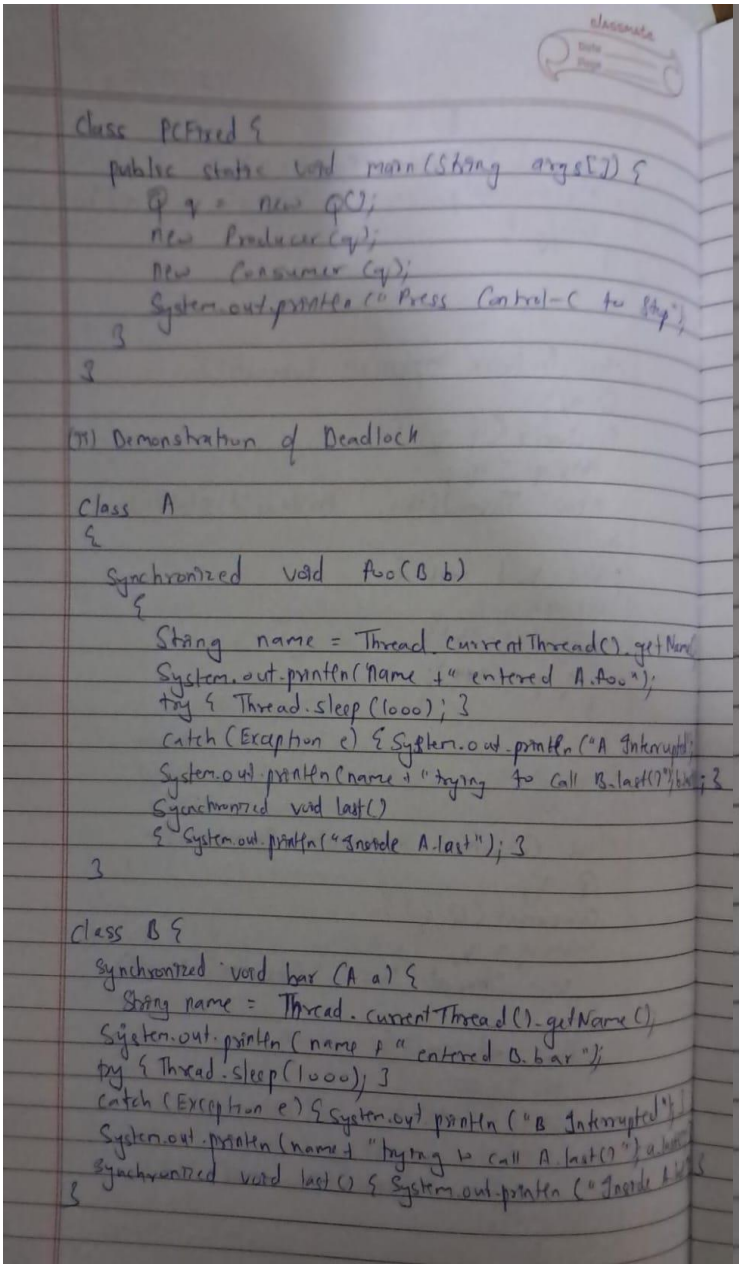
Output

Program 10

Demonstrate Inter process Communication and deadlock.



```
14/12/24

LAB PROGRAM - 10

Demonstrate Interprocess Communication and deadlock
(1) Interprocess Communication:
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("\n Producer 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++;
 }
}
```



Code(i):

```
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.");
}
}
```

```
C:\Users\uzair\OneDrive\Desktop\java\Java>javac PCFixed.java

C:\Users\uzair\OneDrive\Desktop\java\Java>java PCFixed
Press Control-C to stop.
Put: 0

Intimate Consumer


Producer waiting

Got: 0

Intimate Producer

Put: 1

Intimate Consumer

consumed:0

Producer waiting

Got: 1

Intimate Producer

consumed:1
Put: 2

Intimate Consumer


Producer waiting

Got: 2

Intimate Producer

consumed:2
Put: 3

Intimate Consumer
```

Output

Code(ii):

```
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(); }
}
```

```
C:\Users\uzair\OneDrive\Desktop\java\Java\Experiment_10_ii>java Deadlock
MainThread entered A.foo
RacingThread entered B.bar
MainThread trying to call B.last()
RacingThread trying to call A.last()
^C
C:\Users\uzair\OneDrive\Desktop\java\Java\Experiment_10_ii>
```

Output