VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

HRISHIKESH R PRASAD(1BM23CS367)

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 **HRISHIKESH R PRASAD(1BM23CS367),** 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

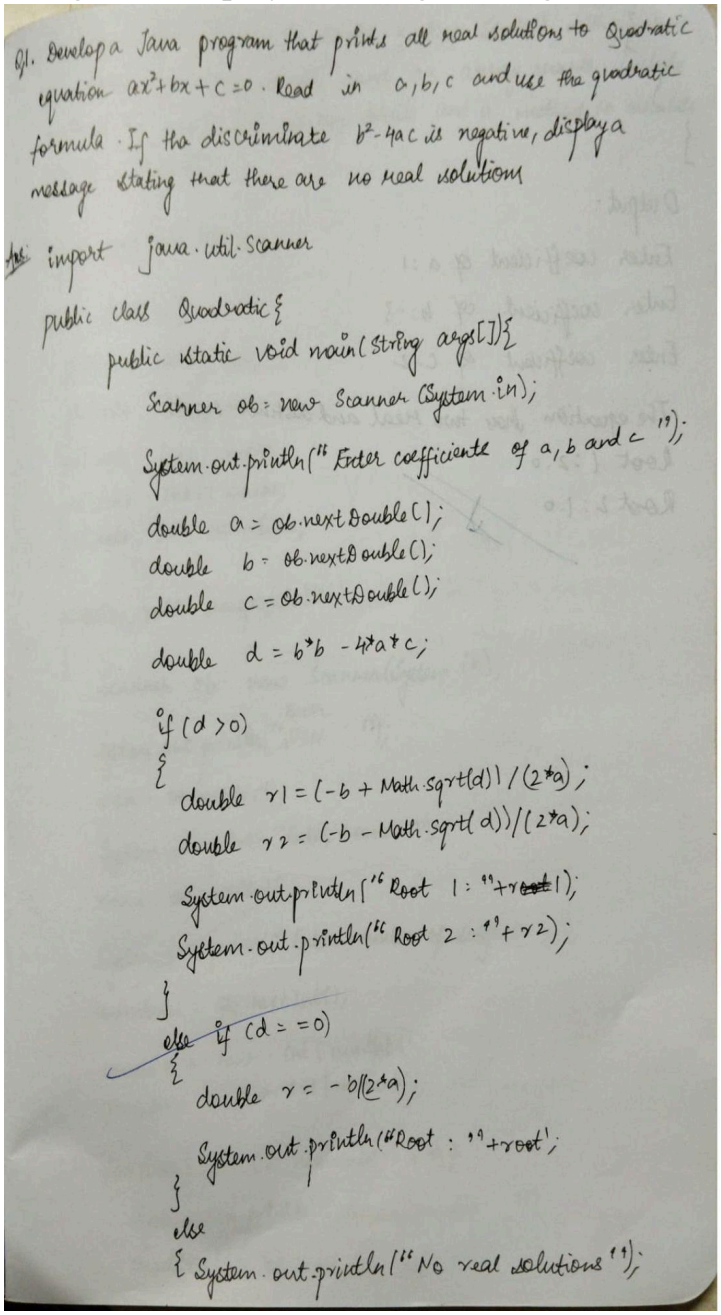
Dr. Kavitha Sooda

Professor and Head,
Department of CSE
BMSCE, Bengaluru

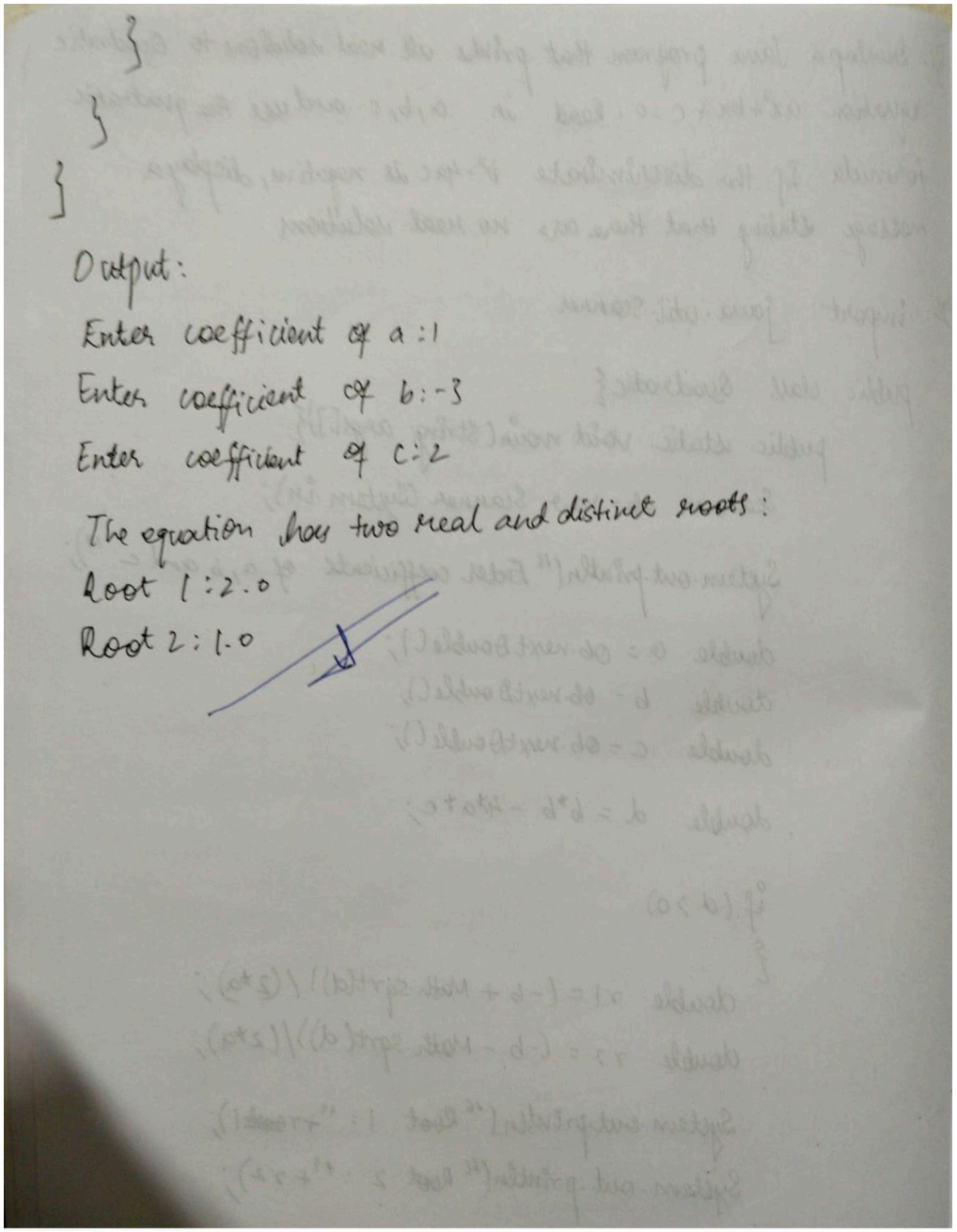
INDEX

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.



```
Q1. Develop a Java program that prints all real solutions to 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

import java.util.Scanner
public class Quadratic {
    public static void main(String args[]){
        Scanner ob = new Scanner(System.in);
        System.out.println("Enter coefficients of a, b and c");
        double a = ob.nextDouble();
        double b = ob.nextDouble();
        double c = ob.nextDouble();
        double d = b*b - 4*a*c;
        if (d > 0)
        {
            double r1 = (-b + Math.sqrt(d)) / (2*a);
            double r2 = (-b - Math.sqrt(d)) / (2*a);
            System.out.println("Root 1 : "+r1);
            System.out.println("Root 2 : "+r2);
        }
        else if (d == 0)
        {
            double r = -b/(2*a);
            System.out.println("Root : "+r);
        }
        else
        { System.out.println("No real solutions");
```




```
import java.util.Scanner;
public class QuadraticEquationSolver { public static void main(String[] args) { Scanner scanner = new
Scanner(System.in);
   // Read coefficients a, b, c from the user
    System.out.print("Enter coefficient a: ");
    double a = scanner.nextDouble();
    System.out.print("Enter coefficient b: ");
    double b = scanner.nextDouble();
    System.out.print("Enter coefficient c: ");
    double c = scanner.nextDouble();

    // Calculate the discriminant: b^2 - 4ac
    double discriminant = b * b - 4 * a * c;

    // Check the nature of the discriminant
    if (discriminant > 0) {
        // Two real and distinct roots
```

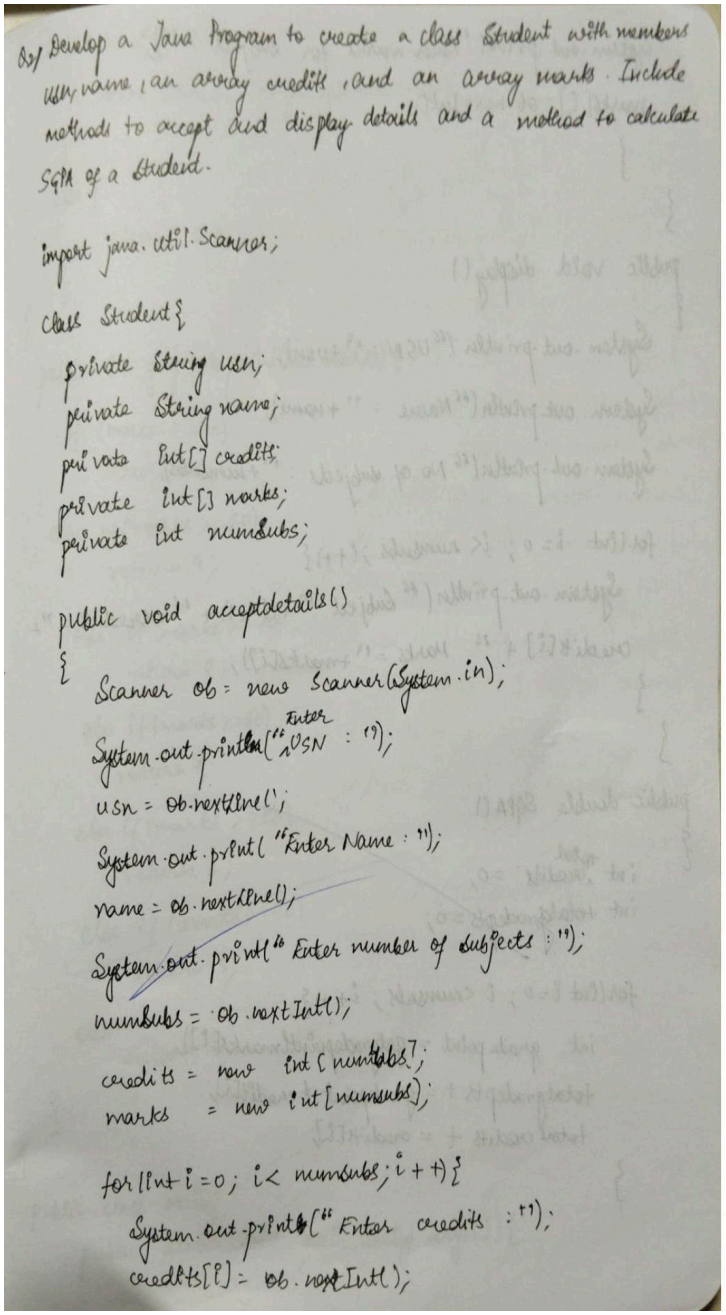
```
        double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
        double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);
        System.out.println("The equation has two real and distinct roots:");
        System.out.println("Root 1: " + root1);
        System.out.println("Root 2: " + root2);
    } else if (discriminant == 0) {
        // One real root (repeated)
        double root = -b / (2 * a);
        System.out.println("The equation has one real and repeated root:");
        System.out.println("Root: " + root);
    } else {
        // No real solutions
        System.out.println("The equation has no real solutions.");
    }

    scanner.close();
}

}
```

OUTPUT

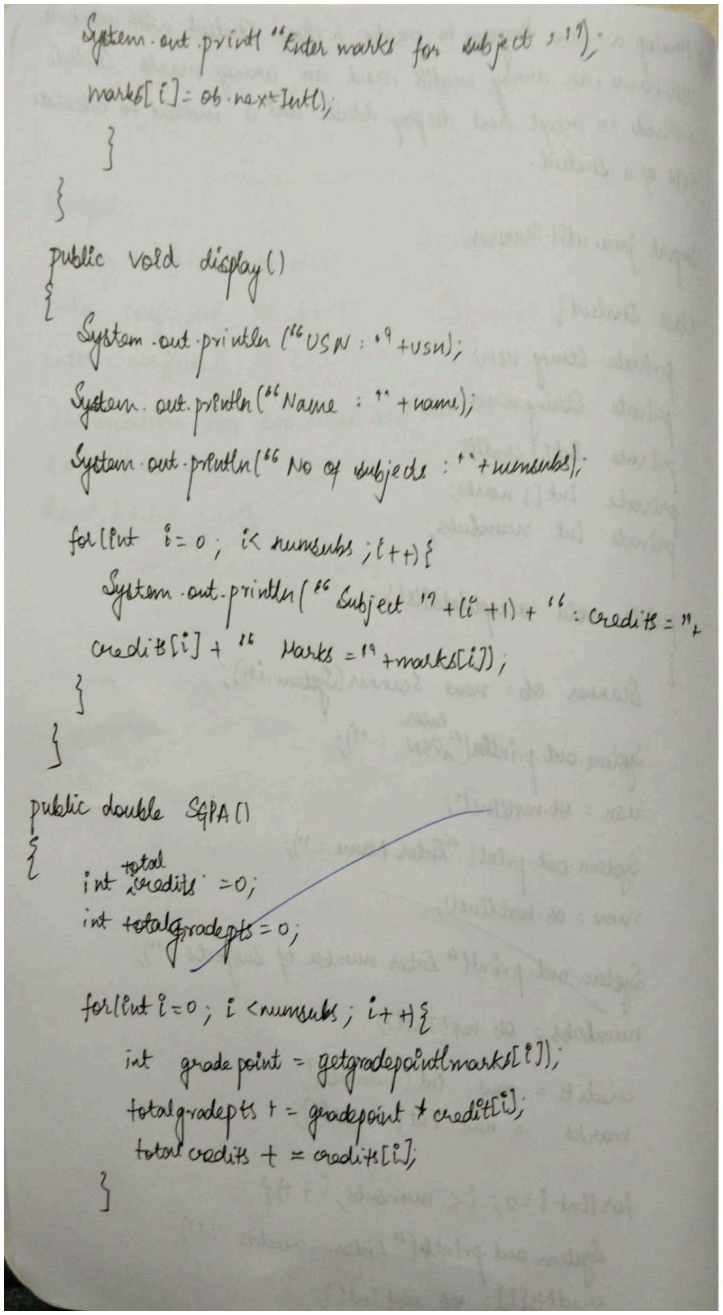
```
Usage: javac <options> <source files>
use --help for a list of possible options
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week1> cd "c:\Users\HP
a\Week1\" ; if ($?) { javac QuadraticEquationSolver.java } ; if ($?) { java
Enter coefficient a: 2
Enter coefficient b: 3
Enter coefficient c: 4
The equation has no real solutions.
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week1> cd "c:\Users\HP
a\Week1\" ; if ($?) { javac QuadraticEquationSolver.java } ; if ($?) { java
Enter coefficient a: 1
Enter coefficient b: -3
Enter coefficient c: 2
The equation has two real and distinct roots:
Root 1: 2.0
Root 2: 1.0
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week1> cd "c:\Users\HP
a\Week1\" ; if ($?) { javac QuadraticEquationSolver.java } ; if ($?) { java
Enter coefficient a: 4
Enter coefficient b: 4
Enter coefficient c: 1
The equation has one real and repeated root:
Root: -0.5
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week1>
```

LABORATORY PROGRAM - 2

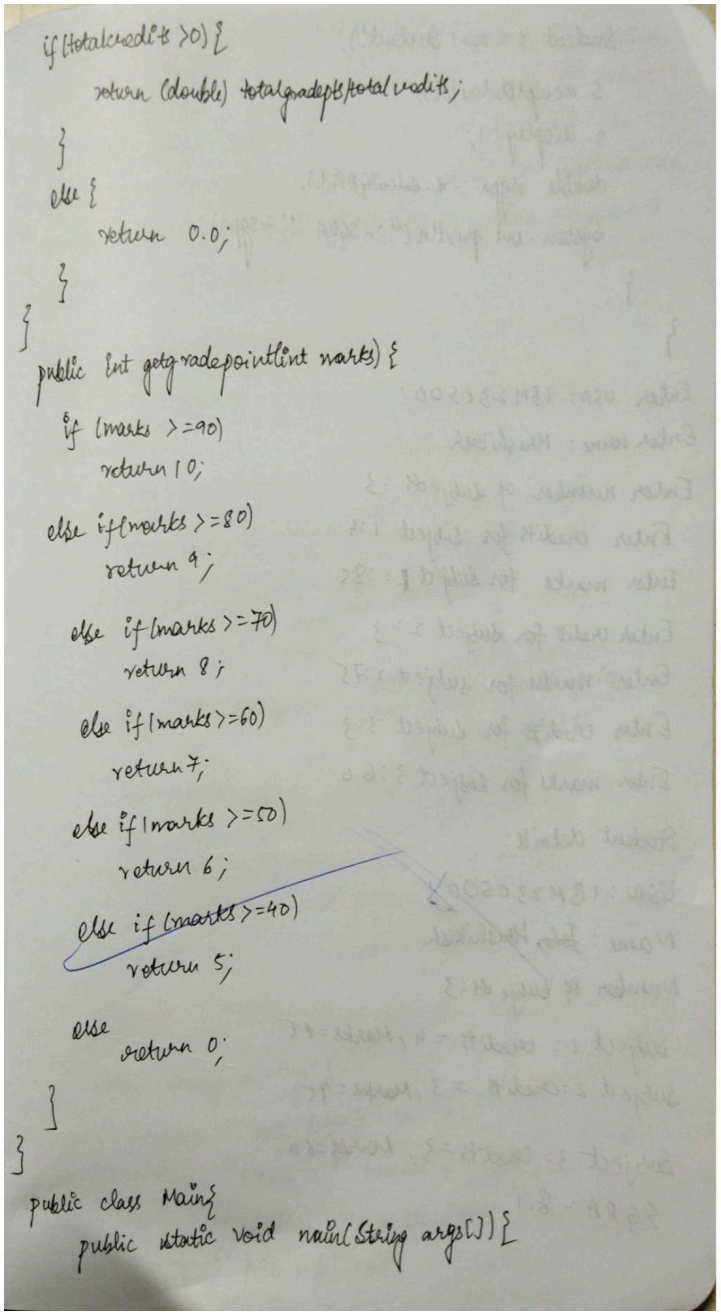
Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.



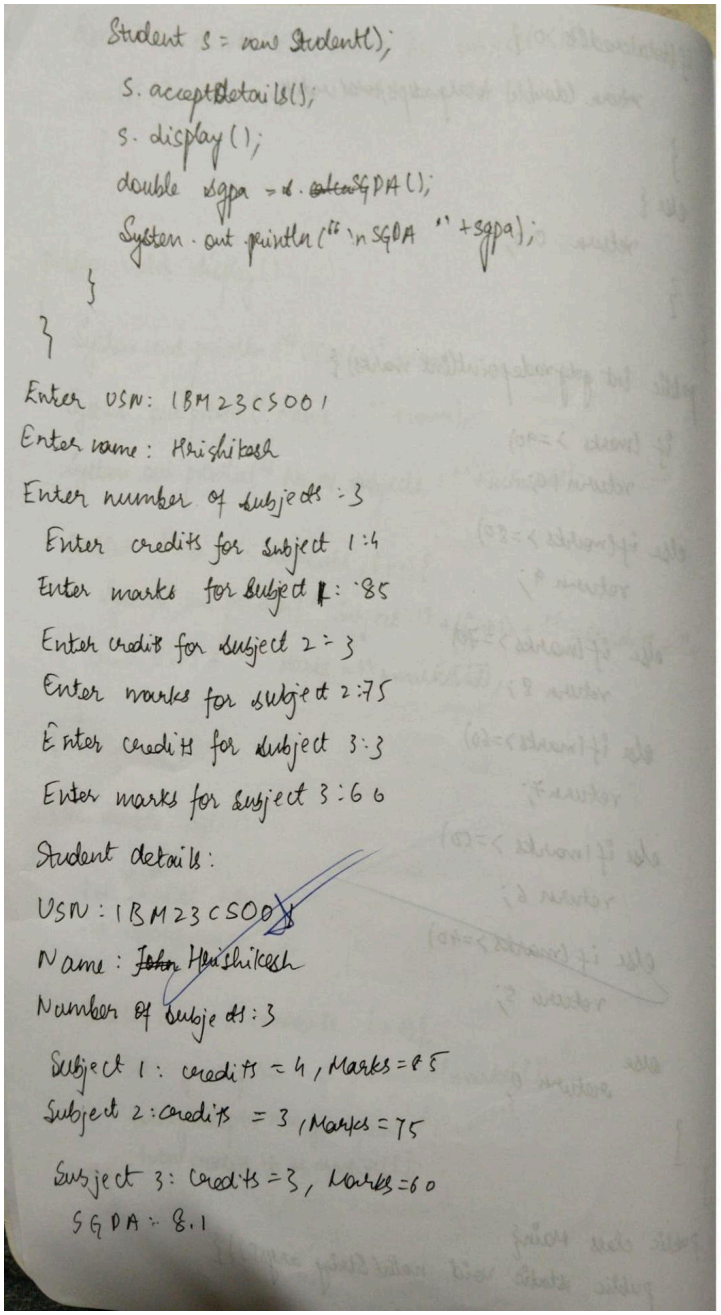
```
Q2) Develop a Java Program to create a class Student with members
usn, name, an array credits, and an array marks. Include
methods to accept and display details and a method to calculate
SGPA of a Student.

import java.util.Scanner;

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

  public void acceptdetails()
  {
    Scanner ob = new Scanner(System.in);
    System.out.println("Enter USN : ");
    usn = ob.nextLine();
    System.out.print("Enter Name : ");
    name = ob.nextLine();
    System.out.print("Enter number of subjects : ");
    numSubs = ob.nextInt();
    credits = new int[numSubs];
    marks = new int[numsubs];

    for(int i=0; i< numsubs; i++){
      System.out.print("Enter credits : ");
      credits[i] = ob.nextInt();
```



```
        System.out.println("Enter marks for subject :"+(i+1));
        marks[i] = ob.nextInt();
      }
  }
  public void display()
  {
    System.out.println("USN : "+usn);
    System.out.println("Name : "+name);
    System.out.println("No of subjects : "+numsubs);
    for(int i=0; i<numsubs;i++){
      System.out.println("Subject "+(i+1)+" : credits = "+
    credits[i] + " Marks = "+marks[i]);
    }
  }
public double SGPA()
{
    int totalcredits =0;
    int totalgradepts = 0;
    for(int i=0; i<numsubs; i++){
       int gradepoint = getgradepoint(marks[i]);
       totalgradepts += gradepoint * credit[i];
       totalcredits += credits[i];
    }
```



```
if (totalcredits >0) {
    return (double) totalgradepts/totalcredits;
  }
  else {
    return 0.0;
  }
}
public int getgradepoint(int marks) {
  if (marks >=90)
    return 10;
  else if(marks >=80)
    return 9;
  else if(marks >=70)
    return 8;
  else if(marks >=60)
    return 7;
  else if(marks >=50)
    return 6;
  else if(marks >=40)
    return 5;
  else
    return 0;
  }
}
public class Main{
  public static void main(String args[]) {
```




```
import java.util.Scanner;

class Student {

    private String usn;
    private String name;
    private int[] credits;
    private int[] marks;
    private int numSubjects;


    public void acceptDetails() {
        Scanner scanner = new Scanner(System.in);
```

```
    System.out.print("Enter USN: ");
    usn = scanner.nextLine();
    System.out.print("Enter Name: ");
    name = scanner.nextLine();

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


    credits = new int[numSubjects];
    marks = new int[numSubjects];


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


public void displayDetails() {
    System.out.println("\nStudent Details:");
    System.out.println("USN: " + usn);
    System.out.println("Name: " + name);
    System.out.println("Number of Subjects: " + numSubjects);

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


public double calculateSGPA() {
    int totalCredits = 0;
    int totalGradePoints = 0;

    for (int i = 0; i < numSubjects; i++) {
        int gradePoint = getGradePoint(marks[i]);
        totalGradePoints += gradePoint * credits[i];
        totalCredits += credits[i];
    }


    if (totalCredits > 0) {
        return (double) totalGradePoints / totalCredits;
    } else {
        return 0.0;
    }
```

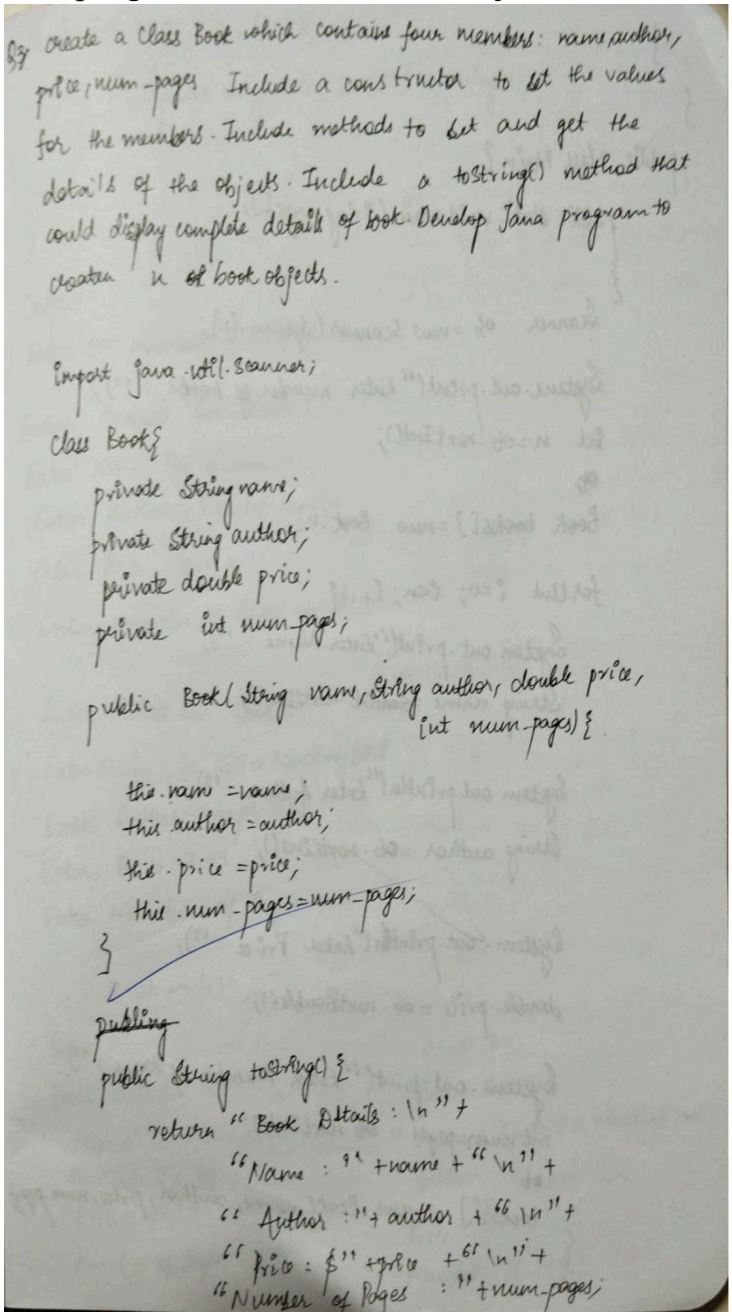
```
    }


    private int getGradePoint(int marks) {
        if (marks >= 90) return 10;
        else if (marks >= 80) return 9;
        else if (marks >= 70) return 8;
        else if (marks >= 60) return 7;
        else if (marks >= 50) return 6;
        else if (marks >= 40) return 5;
        else return 0;
    }
}

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

        Student student = new Student();


        student.acceptDetails();
        student.displayDetails();


        double sgpa = student.calculateSGPA();
        System.out.println("\nSGPA: " + sgpa);
    }
}
```

OUTPUT

```
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava> cd "c:\Users
javac Main.java } ; if ($?) { java Main }
Enter USN: 1BM23CS367
Enter Name: Hrishikesh
Enter the number of subjects: 3
Enter credits for subject 1: 4
Enter marks for subject 1: 85
Enter credits for subject 2: 3
Enter marks for subject 2: 75
Enter credits for subject 3: 3
Enter marks for subject 3: 60

Student Details:
USN: 1BM23CS367
Name: Hrishikesh
Number of Subjects: 3
Subject 1: Credits = 4, Marks = 85
Subject 2: Credits = 3, Marks = 75
Subject 3: Credits = 3, Marks = 60

SGPA: 8.1
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week2>
```

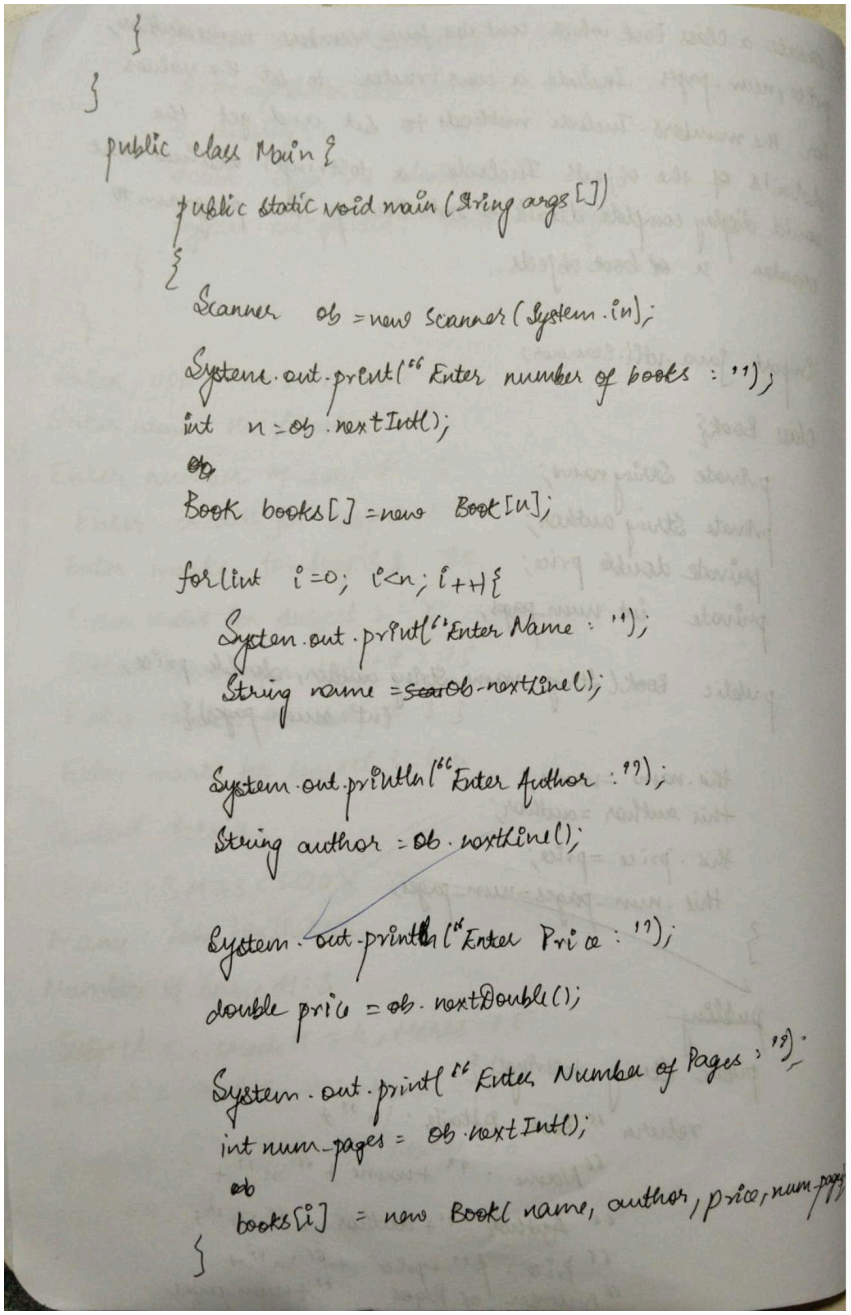
LABORATORY PROGRAM - 3

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.

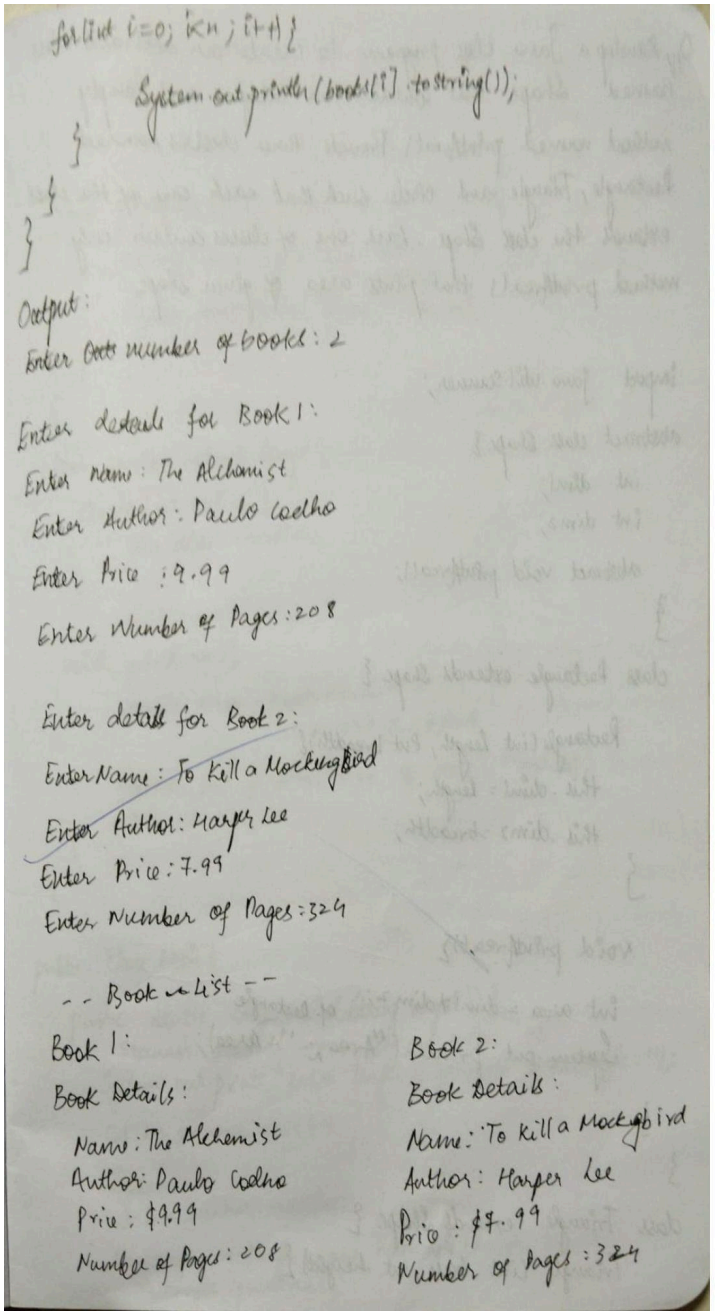


Q3 create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString() method that could display complete details of book. Develop Java program to create n book objects.

```
import java.util.Scanner;
class Book{
    private String name;
    private String author;
    private double price;
    private int num_pages;
    public Book(String name, String author, double price,
                int num_pages){
        this.name = name;
        this.author = author;
        this.price = price;
        this.num_pages = num_pages;
    }
    public String toString(){
        return "Book Details : \n" +
               "Name : " + name + "\n" +
               "Author : " + author + "\n" +
               "Price : $" + price + "\n" +
               "Number of Pages : " + num_pages;
```



```
    }
}
public class Main {
    public static void main (String args[])
    {
      Scanner ob = new Scanner (System.in);
      System.out.print("Enter number of books : ");
      int n = ob.nextInt();
      ob
      Book books[] = new Book[n];
      for(int i=0; i<n; i++){
        System.out.print("Enter Name : ");
        String name = scanOb.nextLine();

        System.out.println("Enter Author :");
        String author = ob.nextLine();

        System.out.println("Enter Price : ");
        double price = ob.nextDouble();

        System.out.print("Enter Number of Pages : ");
        int num_pages = ob.nextInt();
        ob
        books[i] = new Book( name, author, price, num_pag
    }
```




```
import java.util.Scanner;

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

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


public void setDetails(String name, String author, double price, int num_pages) {
```

```
    this.name = name;
    this.author = author;
    this.price = price;
    this.num_pages = num_pages;
}

public String getName() {
    return name;
}

public String getAuthor() {
    return author;
}

public double getPrice() {
    return price;
}

public int getNumPages() {
    return num_pages;
}


public String toString() {
    return "Book Details:\n" +
           "Name: " + name + "\n" +
           "Author: " + author + "\n" +
           "Price: $" + price + "\n" +
           "Number of Pages: " + num_pages;
}


}

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

   System.out.print("Enter the number of books: ");
    int n = scanner.nextInt();
    scanner.nextLine();


    Book[] books = new Book[n];


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

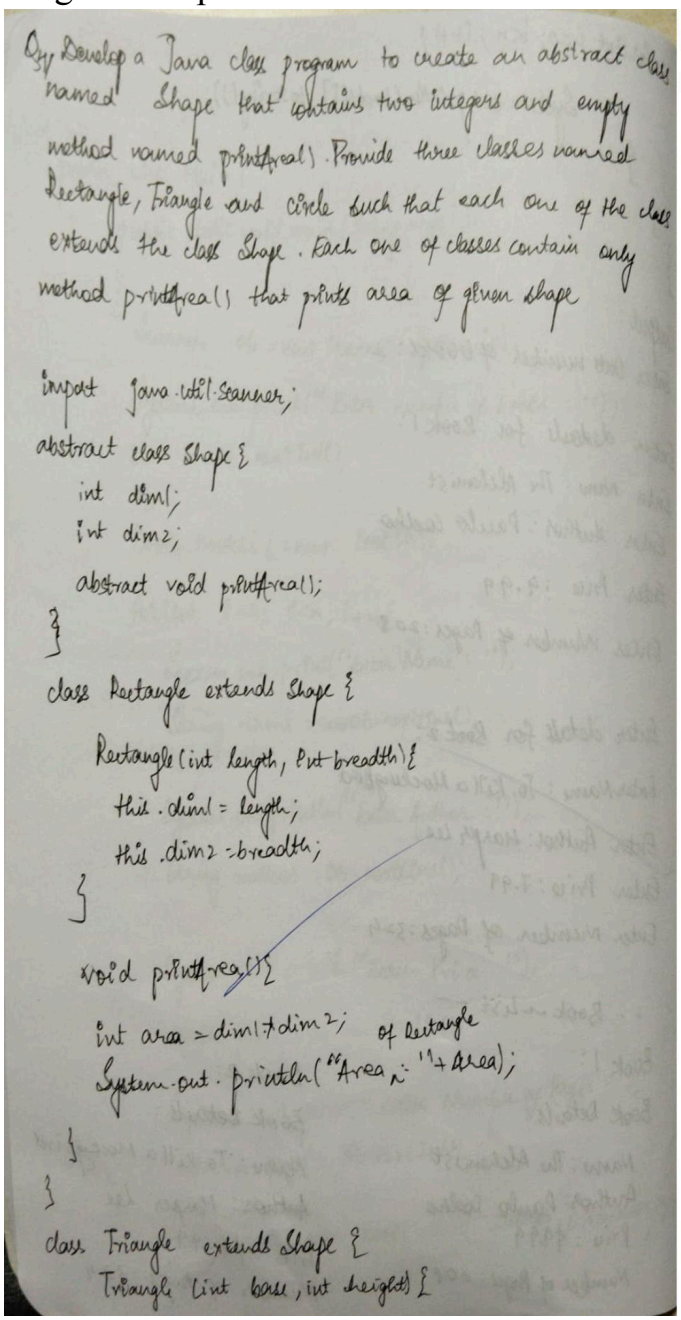
```
        System.out.println("\nEnter details for Book " + (i + 1) + ":");

        System.out.print("Enter Name: ");
        String name = scanner.nextLine();

        System.out.print("Enter Author: ");
        String author = scanner.nextLine();

        System.out.print("Enter Price: ");
        double price = scanner.nextDouble();

        System.out.print("Enter Number of Pages: ");
        int num_pages = scanner.nextInt();
        scanner.nextLine();


        books[i] = new Book(name, author, price, num_pages);
    }


    System.out.println("\n--- Book List ---");
    for (int i = 0; i < n; i++) {
        System.out.println("\nBook " + (i + 1) + ":");
        System.out.println(books[i].toString());
    }

    scanner.close();
}


}
```

OUTPUT

```
$?) { javac Main.java } ; if ($?) { java Main }
Enter the number of books: 2

Enter details for Book 1:
Enter Name: The Alchemist
Enter Author: Paulo Coelho
Enter Price: 9.99
Enter Number of Pages: 208

Enter details for Book 2:
Enter Name: To Kill a Mockingbird
Enter Author: Harper Lee
Enter Price: 7.99
Enter Number of Pages: 324

--- Book List ---

Book 1:
Book Details:
Name: The Alchemist
Author: Paulo Coelho
Price: $9.99
Number of Pages: 208

Book 2:
Book Details:
Name: To Kill a Mockingbird
Author: Harper Lee
Price: $7.99
Number of Pages: 324
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week3>
```

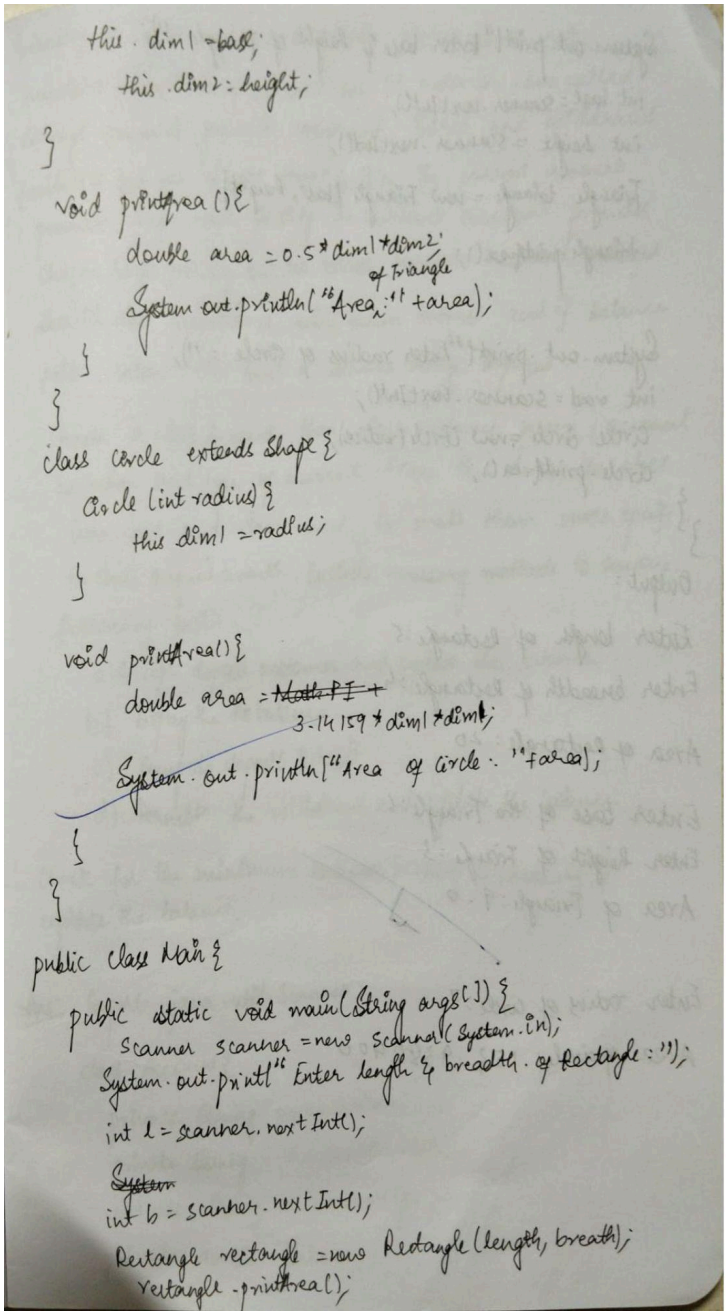
LABORATORY PROGRAM - 4

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.

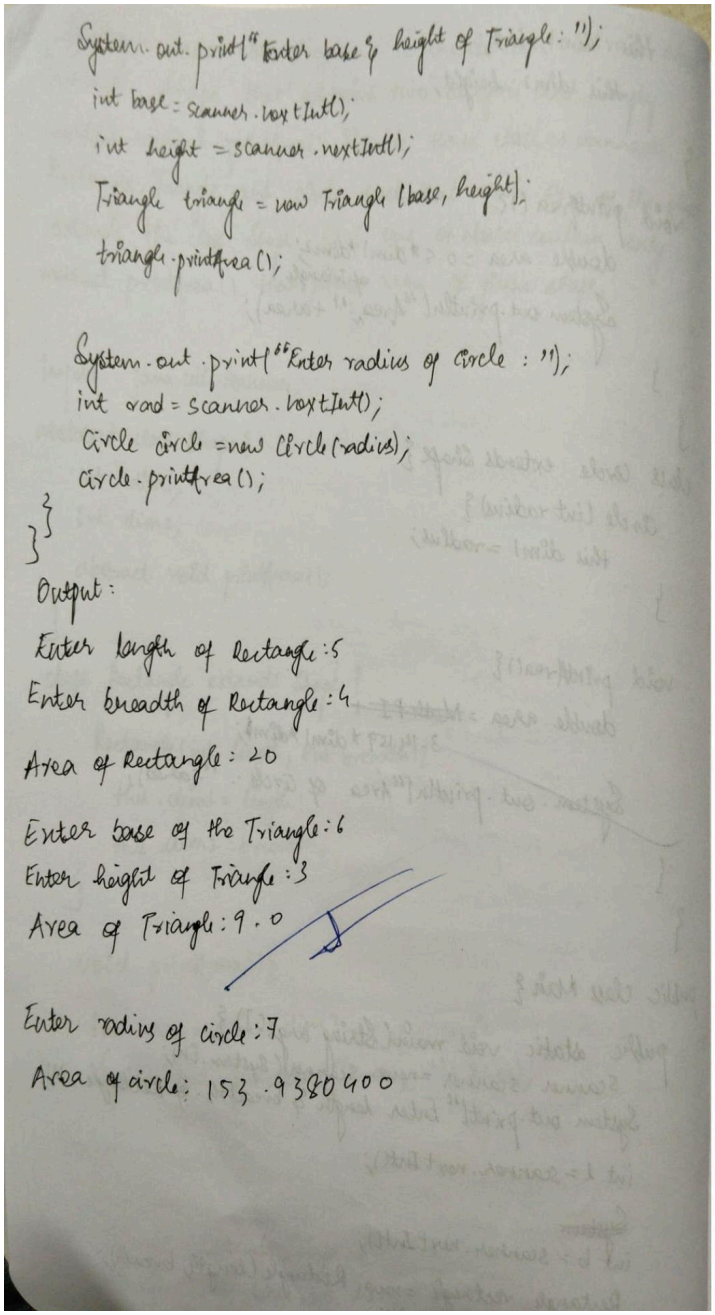


Q) Develop a Java class program to create an abstract class named Shape that contains two integers and empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the class extends the class Shape. Each one of classes contain only method printArea()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

```
import java.util.Scanner;
abstract class Shape {
    int dim1;
    int dim2;
    abstract void printArea();
}
class Rectangle extends Shape {
    Rectangle(int length, int breadth){
      this.dim1 = length;
      this.dim2 = breadth;
    }
    void printArea(){
      int area = dim1*dim2;  of Rectangle
      System.out.println("Area :"+area);
    }
}
class Triangle extends Shape {
    Triangle(int base, int height) {
```



```
    this.dim1 = base;
    this.dim2 = height;
}

void printArea(){
    double area = 0.5 * dim1 * dim2;
    System.out.println("Area of Triangle: " + area);
}
}

class Circle extends Shape {
    Circle(int radius) {
        this.dim1 = radius;
    }

    void printArea() {
        double area = 3.14159 * dim1 * dim1;
        System.out.println("Area of circle: " + area);
    }
}

public class Main {
    public static void main(String args[]) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter length & breadth of Rectangle: ");
        int l = scanner.nextInt();
        int b = scanner.nextInt();
        Rectangle rectangle = new Rectangle(length, breath);
        rectangle.printArea();
```



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

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

Output:

```
Enter length of Rectangle: 5
Enter breadth of Rectangle: 4
Area of Rectangle: 20

Enter base of the Triangle: 6
Enter height of Triangle: 3
Area of Triangle: 9.0

Enter radius of circle: 7
Area of circle: 153.93804002
```

```
import java.util.Scanner;

abstract class Shape { int dimension1; int dimension2;

abstract void printArea();


}

class Rectangle extends Shape {

Rectangle(int length, int breadth) {
   this.dimension1 = length;
   this.dimension2 = breadth;
}
```

```
void printArea() {
    int area = dimension1 * dimension2;
    System.out.println("Area of Rectangle: " + area);
}


}

class Triangle extends Shape {

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

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


}

class Circle extends Shape {

Circle(int radius) {
    this.dimension1 = radius;
}


void printArea() {
    double area = Math.PI * dimension1 * dimension1;
    System.out.println("Area of Circle: " + area);
}


}

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

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


    System.out.print("\nEnter the base of the Triangle: ");
    int base = scanner.nextInt();
    System.out.print("Enter the height of the Triangle: ");
    int height = scanner.nextInt();
```

```
    Triangle triangle = new Triangle(base, height);
    triangle.printArea();


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

    scanner.close();
}


}
```

OUTPUT

```
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week3> cd "c:\Users\HP\Des
$?) { javac Main.java } ; if ($?) { java Main }
Enter the length of the Rectangle: 5
Enter the breadth of the Rectangle: 4
Area of Rectangle: 20

Enter the base of the Triangle: 6
Enter the height of the Triangle: 3
Area of Triangle: 9.0

Enter the radius of the Circle: 7
Area of Circle: 153.93804002589985
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week4>
```

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.

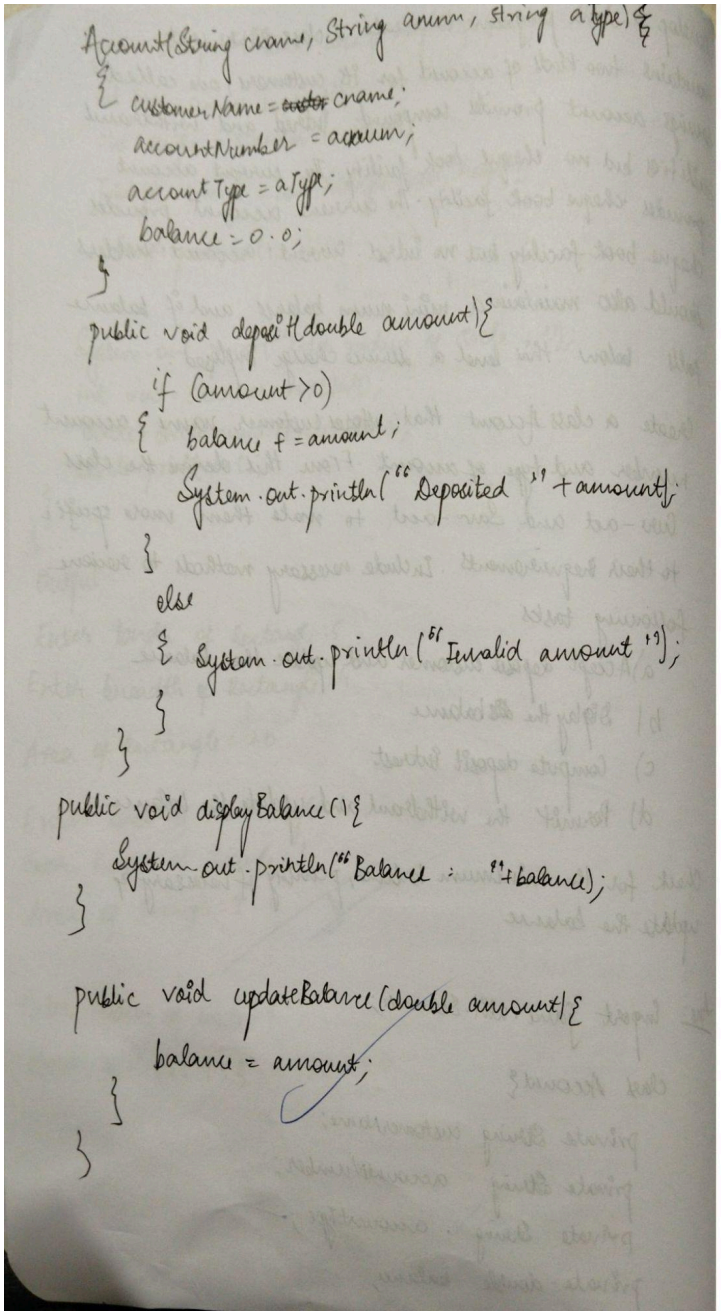
Q. Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility. The current account provides cheque book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

Create a class Account that stores customer name, account number and type of account. From this derive the class Cur-acct and Sav-acct to make them more specific to their requirements. Include necessary methods to receive following tasks

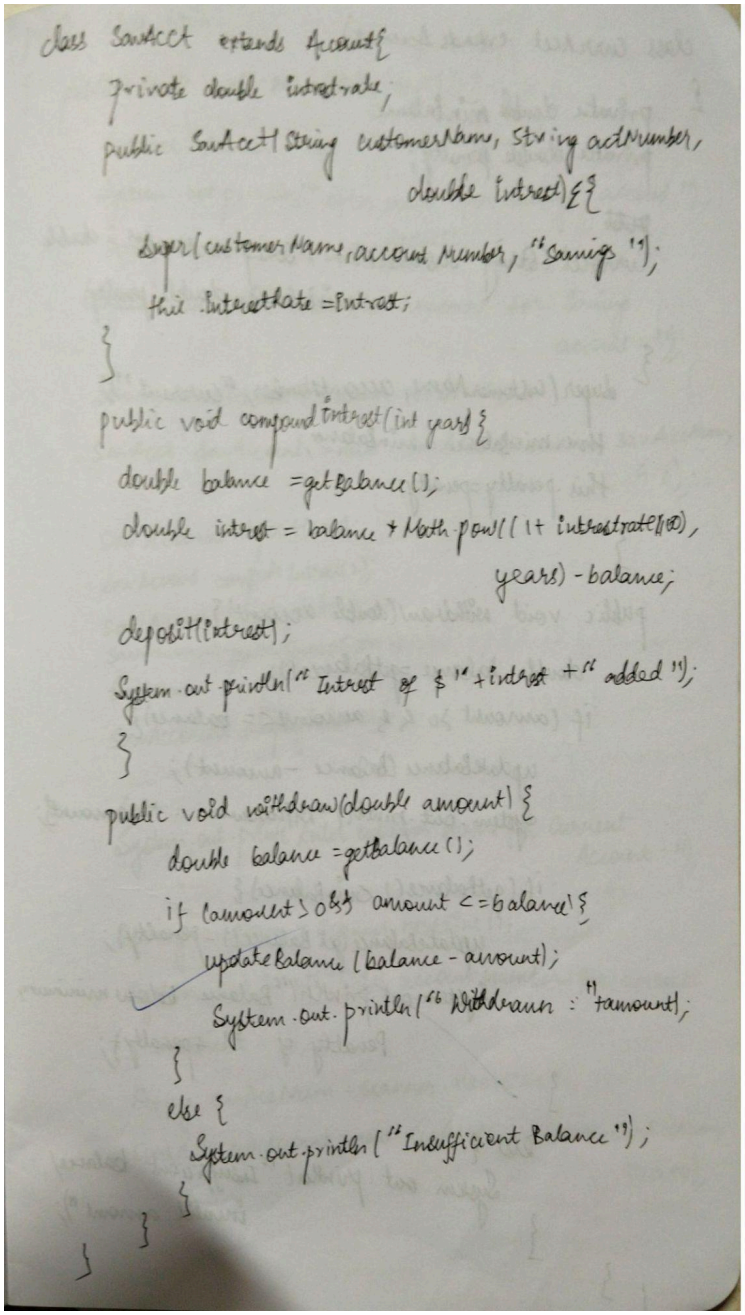
a) Accept deposit customer and update the balance
b) Display the balance
c) Compute deposit interest
d) Permit the withdrawal and update the balance

Check for the minimum balance, penalty if necessary & update the balance

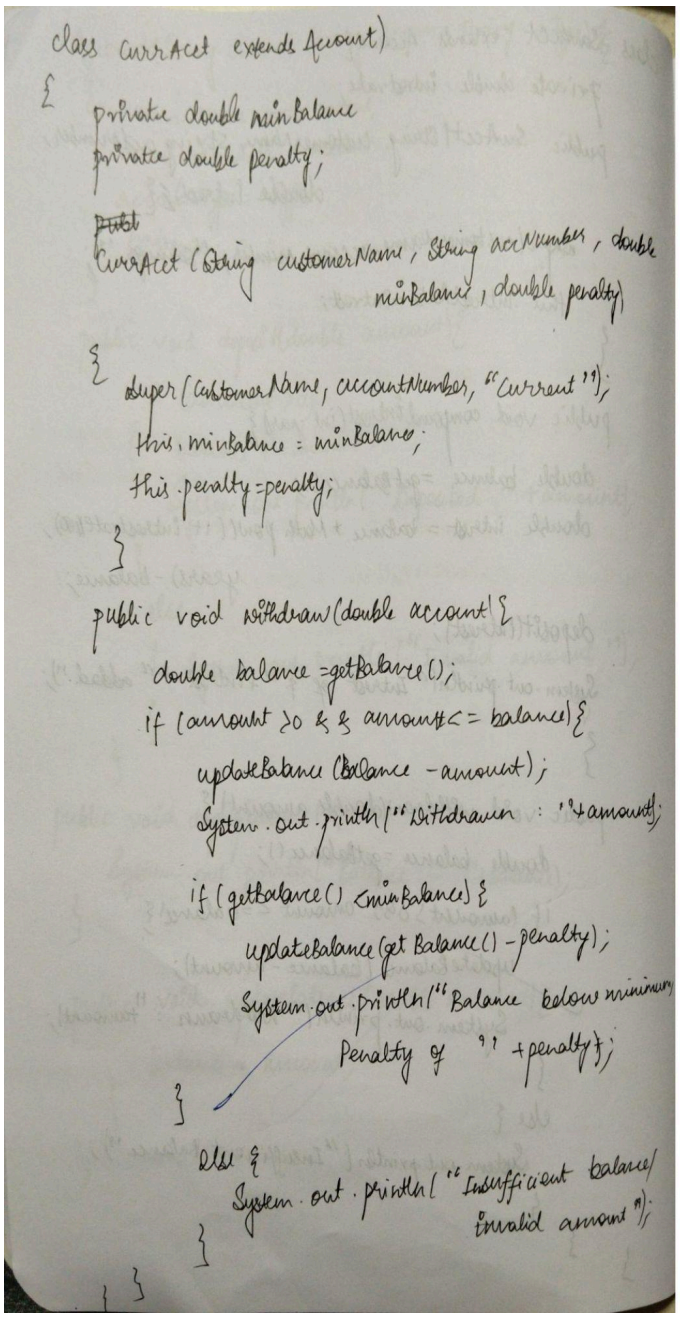
Ans:
```
import java.util.Scanner;
class Account{
    private String customerName;
    private String accountNumber;
    private String accountType;
    private double balance;
```



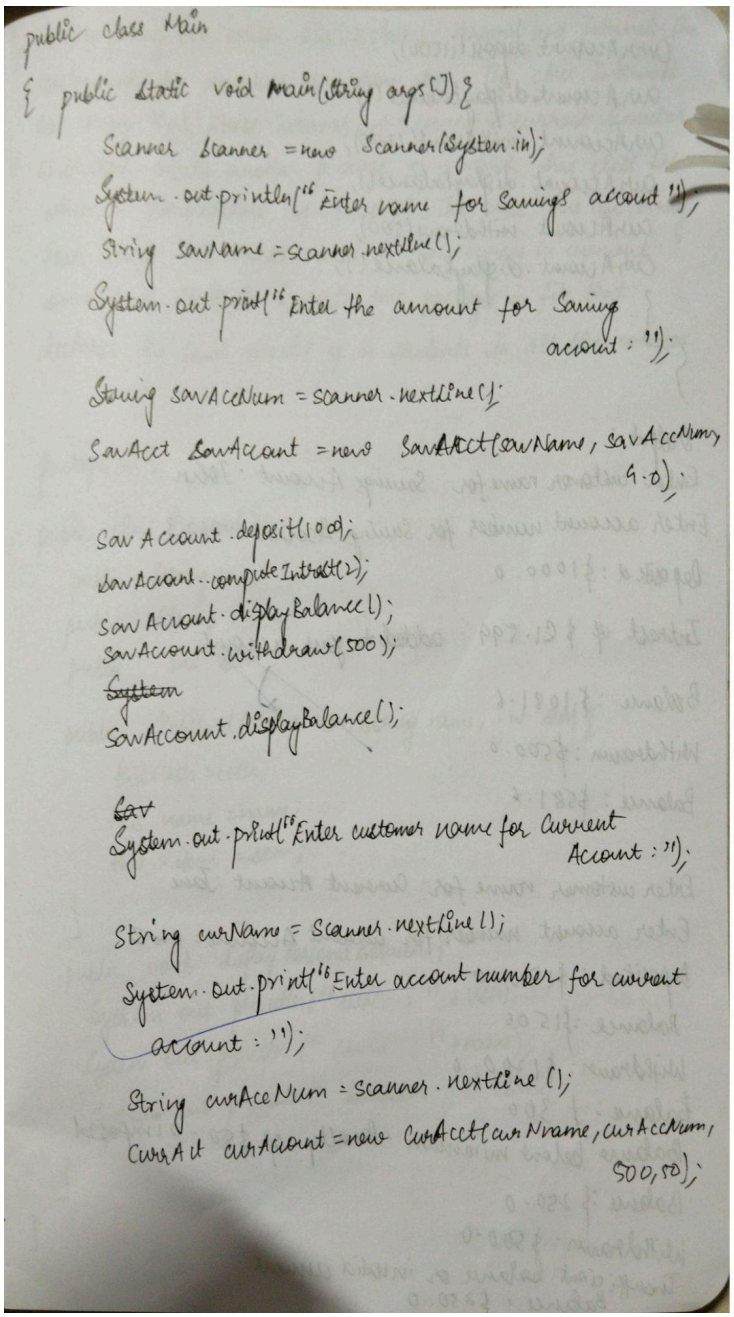
```
Account(String cname, String anum, String atype){
{ customerName = cname;
  accountNumber = anum;
  accountType = atype;
  balance = 0.0;
}
public void deposit(double amount){
    if (amount>0)
    { balance += amount;
      System.out.println("Deposited "+amount);
    }
     else
    { System.out.println("Invalid amount");
    }
  }
public void displayBalance(){
   System.out.println("Balance : "+balance);
}

public void updateBalance(double amount){
     balance = amount;
  }
}
```



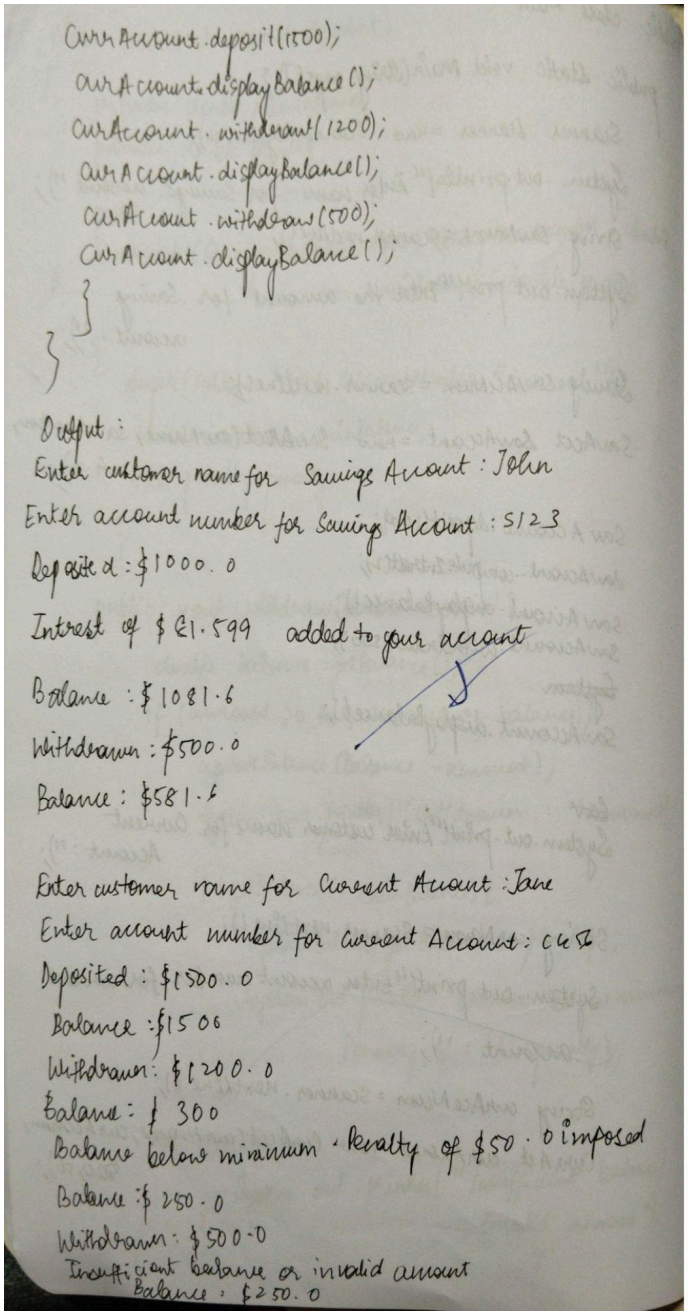
```
class SavAcct extends Account{
    private double intrestrate;
    public SavAcct(String customerName, String actNumber,
                   double Intrest){}
      super(customerName, account number, "Savings");
      this.Interestrate = Intrest;
    }
    public void compoundInterest(int years){
      double balance = getBalance();
      double interest = balance * Math.pow((1+ interestrate/100),
                                           years) - balance;
      deposit(interest);
      System.out.println("Interest of $" +intrest + " added");
    }
    public void withdraw(double amount){
        double balance = getbalance();
        if (amount > 0 && amount <= balance){
          updateBalance(balance - amount);
          System.out.println("Withdrawn : "+amount);
        }
        else {
          System.out.println("Insufficient Balance");
        }
    }
}
```



```
class CurrAcct extends Account)
{
    private double minBalance
    private double penalty;

    public
    CurrAcct (String customerName, String accNumber, double
                    minBalance, double penalty)

    {
        super (customerName, accountNumber, "Current");
        this.minBalance = minBalance;
        this.penalty = penalty;
    }

public void withdraw(double account) {
    double balance = getBalance();
    if (amount > 0 && amount <= balance) {
        updateBalance (Balance - amount);
        System.out.println("Withdrawn : "+amount);

        if (getBalance() < minBalance) {
            updateBalance(get Balance() - penalty);
            System.out.println("Balance below minimum
                    Penalty of " +penalty);
        }

        else {
            System.out.println("Insufficient balance/
                    Invalid amount");
        }
    }
}
```



```
public class Main
{ public static void main(String args[]) {
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter name for Savings account");
    String savName = scanner.nextLine();
    System.out.printf("Enter the amount for Saving
                                    account :");
    String savAccNum = scanner.nextLine();
    SavAcct SavAccount = new SavAcct(savName, savAccNum,
                                    9.0);

    SavAccount.deposit(100);
    SavAccount.computeInterest(2);
    SavAccount.displayBalance();
    SavAccount.withdraw(500);
    System
    SavAccount.displayBalance();

    Sav
    System.out.printf("Enter customer name for Current
                                    Account :");

    String curName = Scanner.nextLine();
    System.out.printf("Enter account number for current
    account :");
    String curAccNum = scanner.nextLine();
    CurrAct curAccount = new CurrAct(curName, curAccNum,
                                    500,50);
```




```
import java.util.Scanner;

class Account {
    private String customerName;
    private String accountNumber;
    private String accountType;
    private double balance;


    public Account(String customerName, String accountNumber, String accountType) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = 0.0;
    }


    public String getCustomerName() {
```

```
        return customerName;
    }

    public String getAccountNumber() {
        return accountNumber;
    }
    public String getAccountType() {
        return accountType;
    }

    public double getBalance() {
        return balance;
    }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited: $" + amount);
        } else {
            System.out.println("Invalid deposit amount.");
        }
    }

    public void displayBalance() {
        System.out.println("Balance: $" + balance);
    }

    public void updateBalance(double amount) {
        balance = amount;
    }
}

class SavAcct extends Account {
    private double interestRate;

    public SavAcct(String customerName, String accountNumber, double interestRate) {
        super(customerName, accountNumber, "Savings");
        this.interestRate = interestRate;
    }

    public void computeInterest(int years) {
        double balance = getBalance();
        double interest = balance * Math.pow((1 + interestRate / 100), years) - balance;
        deposit(interest);
        System.out.println("Interest of $" + interest + " added to your account.");
    }

    public void withdraw(double amount) {
        double balance = getBalance();
        if (amount > 0 && amount <= balance) {
            updateBalance(balance - amount);
            System.out.println("Withdrawn: $" + amount);
        } else {
            System.out.println("Insufficient balance or invalid amount.");
        }
```

```
    }
}

class CurAcct extends Account {
    private double minBalance;
    private double penalty;


    public CurAcct(String customerName, String accountNumber, double minBalance, double penalty) {
        super(customerName, accountNumber, "Current");
        this.minBalance = minBalance;
        this.penalty = penalty;
    }


    public void withdraw(double amount) {
        double balance = getBalance();
        if (amount > 0 && amount <= balance) {
            updateBalance(balance - amount);
            System.out.println("Withdrawn: $" + amount);


            if (getBalance() < minBalance) {
                updateBalance(getBalance() - penalty);
                System.out.println("Balance below minimum. Penalty of $" + penalty + " imposed.");
            }
        } else {
            System.out.println("Insufficient balance or invalid amount.");
        }
    }
}

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


        System.out.print("Enter customer name for Savings Account: ");
        String savName = scanner.nextLine();
        System.out.print("Enter account number for Savings Account: ");
        String savAccNum = scanner.nextLine();
        SavAcct savAccount = new SavAcct(savName, savAccNum, 4.0);

        savAccount.deposit(1000);
        savAccount.computeInterest(2);
        savAccount.displayBalance();
        savAccount.withdraw(500);
        savAccount.displayBalance();

        System.out.println("\n--------------------\n");


        System.out.print("Enter customer name for Current Account: ");
        String curName = scanner.nextLine();
        System.out.print("Enter account number for Current Account: ");
        String curAccNum = scanner.nextLine();
        CurAcct curAccount = new CurAcct(curName, curAccNum, 500, 50);
        curAccount.deposit(1500);
        curAccount.displayBalance();
        curAccount.withdraw(1200);
        curAccount.displayBalance();
        curAccount.withdraw(500);
```

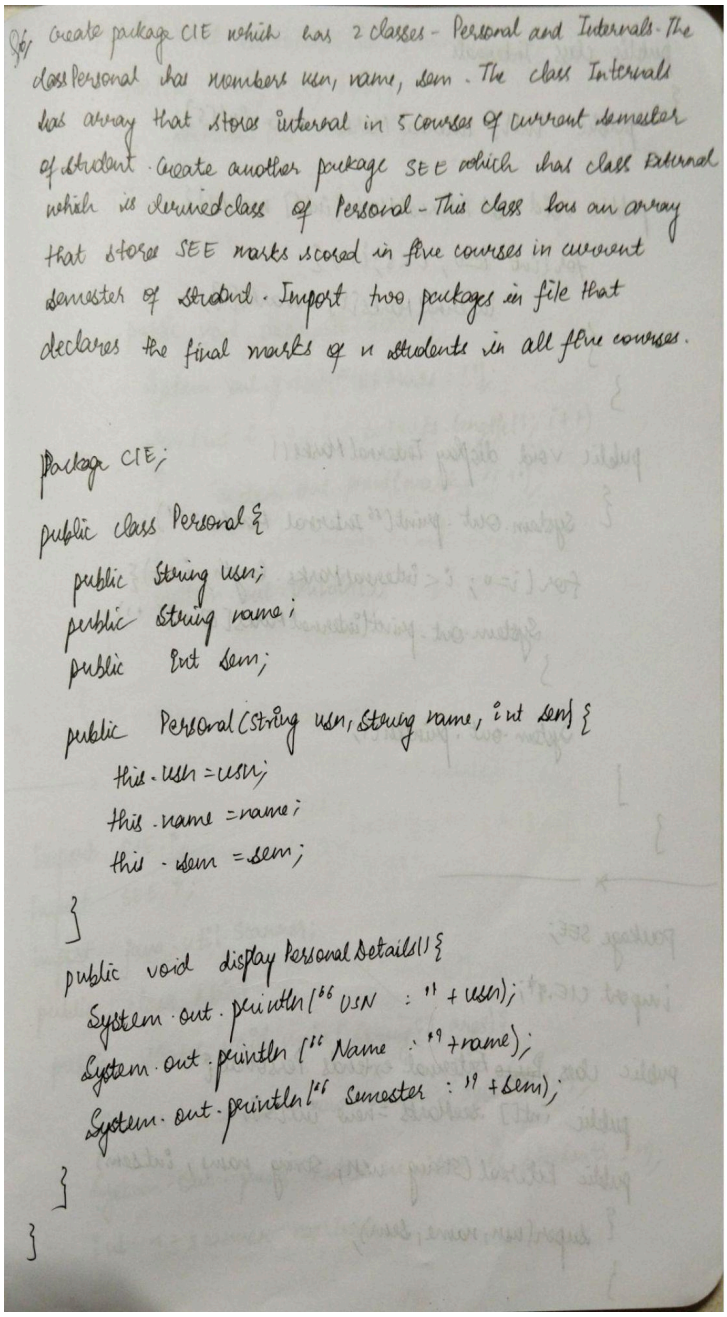
```
        curAccount.displayBalance();

        scanner.close();
    }
}
```

OUTPUT

```
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week4> cd "c:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week5\" ; if ($?) { javac Main.java } ; if ($?) { java Main }
Enter customer name for Savings Account: John
Enter account number for Savings Account: s123
Deposited: $1000.0
Deposited: $81.60000000000014
Interest of $81.60000000000014 added to your account.
Balance: $1081.6000000000001
Withdrawn: $500.0
Balance: $581.6000000000001

--------------------
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week4> cd "c:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week5\" ; if ($?) { javac Main.java } ; if ($?) { java Main }
Enter customer name for Savings Account: John
Enter account number for Savings Account: s123
Deposited: $1000.0
Deposited: $81.60000000000014
Interest of $81.60000000000014 added to your account.
Balance: $1081.6000000000001
Withdrawn: $500.0
Balance: $581.6000000000001

--------------------

Enter customer name for Current Account: Jane
Enter account number for Current Account: c456
Deposited: $1500.0
Enter customer name for Savings Account: John
Enter account number for Savings Account: s123
Deposited: $1000.0
Deposited: $81.60000000000014
Interest of $81.60000000000014 added to your account.
Balance: $1081.6000000000001
Withdrawn: $500.0
Balance: $581.6000000000001

--------------------

--------------------

Enter customer name for Current Account: Jane
--------------------

Enter customer name for Current Account: Jane

Enter customer name for Current Account: Jane
Enter account number for Current Account: c456
Deposited: $1500.0
Balance: $1500.0
Enter account number for Current Account: c456
Deposited: $1500.0
Balance: $1500.0
Withdrawn: $1200.0
Balance below minimum. Penalty of $50.0 imposed.
Balance: $250.0
Insufficient balance or invalid amount.
Balance: $250.0
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week5>
```

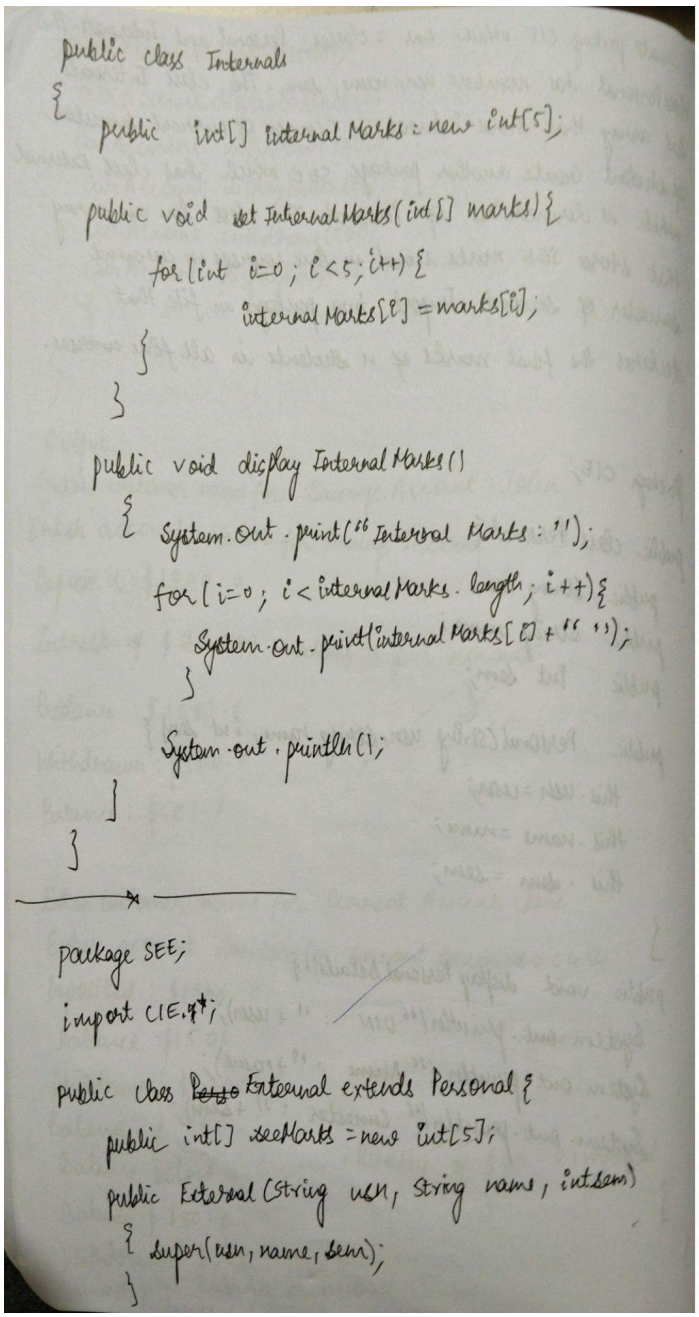
LABORATORY PROGRAM - 6

Create a package CIE which has two classes - Personal 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 Personal. 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.



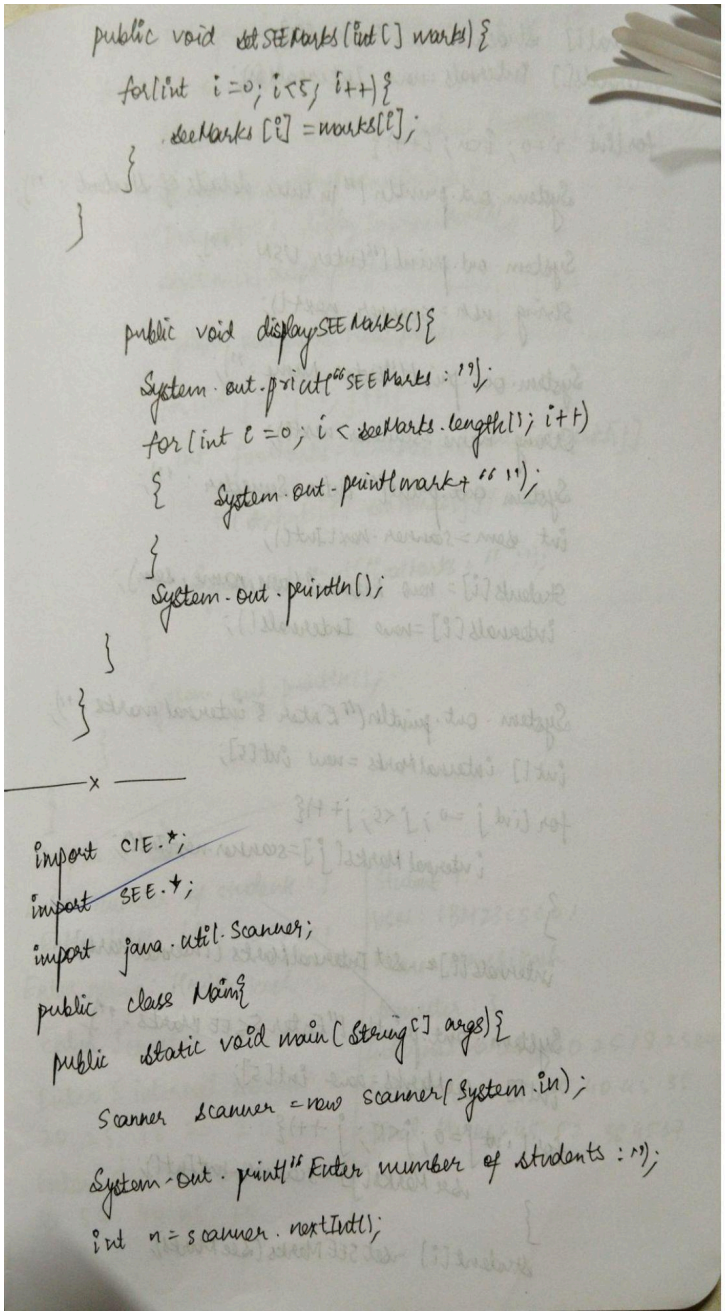
Q6) Create package CIE which has 2 classes - Personal and Internals. The class Personal has members usn, name, sem. The class Internals has array that stores internal in 5 courses of current semester of student. Create another package SEE which has class External which is derived class of Personal. This class has an array that stores SEE marks scored in five courses in current semester of student. Import two packages in file that declares the final marks of n students in all five courses.

```
package CIE;
public class Personal {
    public String usn;
    public String name;
    public int sem;
    public Personal(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }
    public void displayPersonalDetails() {
        System.out.println("USN : " + usn);
        System.out.println("Name : " + name);
        System.out.println("Semester : " + sem);
    }
}
```



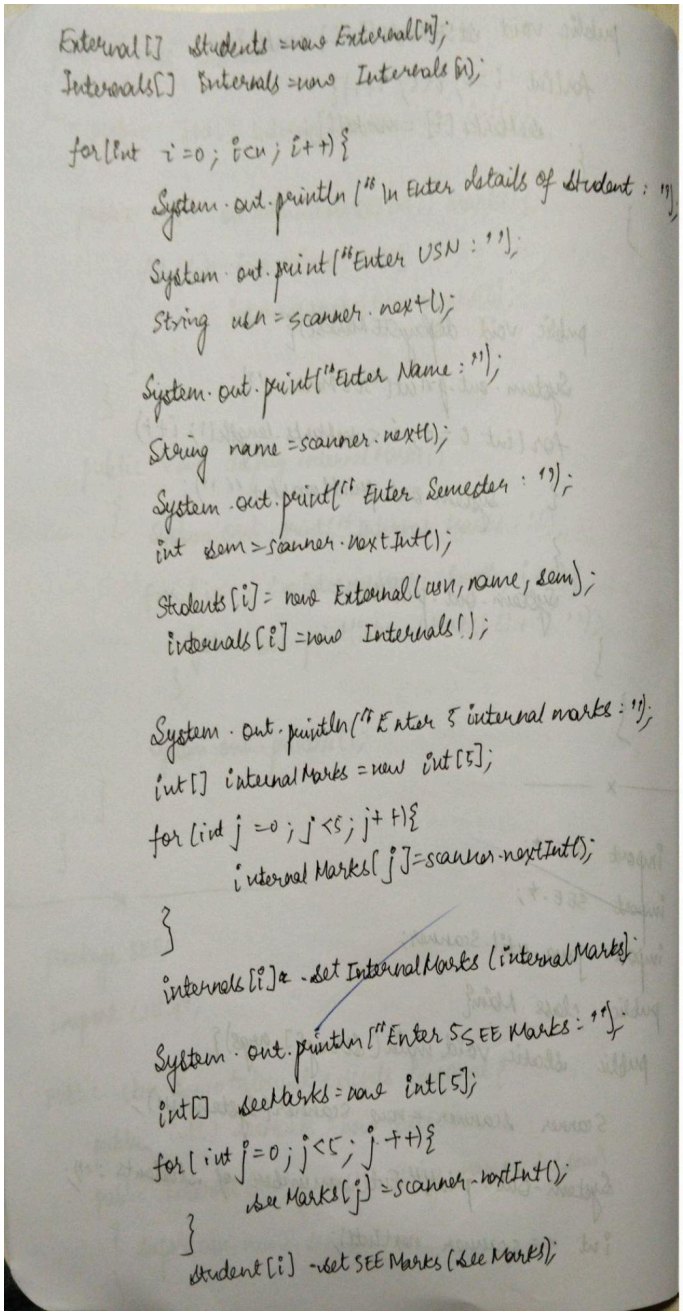
```
public class Internals
{
    public int[] internalMarks = new int[5];

    public void setInternalMarks(int[] marks) {
        for(int i=0; i<5; i++) {
            internalMarks[i] = marks[i];
        }
    }

    public void displayInternalMarks()
    {
        System.out.print("Internal Marks: ");
        for(i=0; i<internalMarks.length; i++){
            System.out.print(internalMarks[i] + " ");
        }

        System.out.println();
    }
}
```

```
package SEE;
import CIE.*;

public class ~~Resu~~ External extends Personal {
    public int[] seeMarks = new int[5];
    public External(String usn, String name, int sem)
    {
        super(usn, name, sem);
    }
```

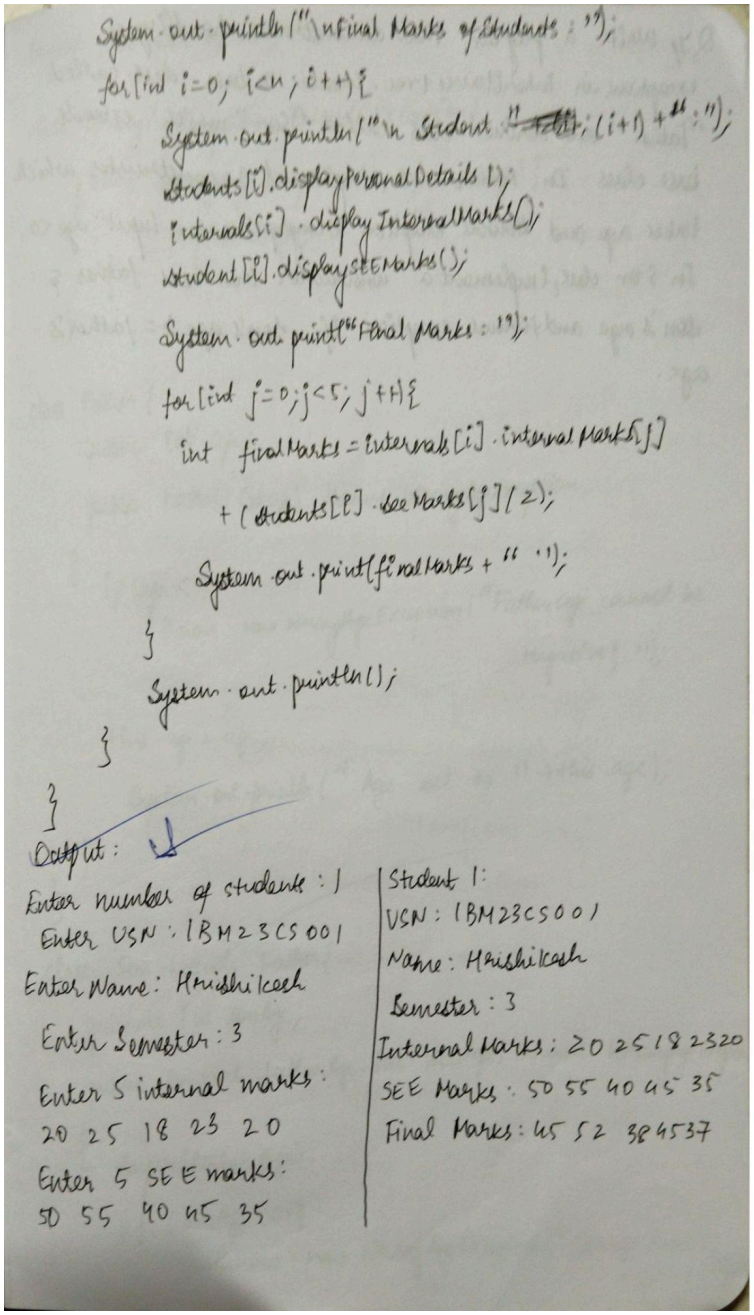


```
public void setSEEMarks(int[] marks){
    for(int i=0; i<5; i++){
        seeMarks[i] = marks[i];
    }
}

public void displaySEEMarks(){
    System.out.println("SEE Marks : ");
    for(int i=0; i < seeMarks.length[1]; i++)
    {   System.out.println(mark+ " ");
    }
    System.out.println();
}
}
```

```
import CIE.*;
import SEE.*;
import java.util.Scanner;
public class Main{
  public static void main(String[] args){
    Scanner scanner = new scanner(System.in);
    System.out.println("Enter number of students : ");
    int n = scanner.nextInt();
```



```
External[] students = new External[n];
Internals[] internals = new Internals[n];

for(int i=0; i<n; i++){
    System.out.println("\n Enter details of student : ");
    System.out.print("Enter USN : ");
    String usn = scanner.next();
    System.out.print("Enter Name : ");
    String name = scanner.next();
    System.out.print("Enter Semester : ");
    int sem = scanner.nextInt();
    students[i] = new External(usn, name, sem);
    internals[i] = new Internals();

    System.out.println("Enter 5 internal marks : ");
    int[] internalMarks = new int[5];
    for(int j=0; j<5; j++){
        internalMarks[j] = scanner.nextInt();
    }
    internals[i].setInternalMarks(internalMarks);
    System.out.println("Enter 5 SEE Marks : ");
    int[] seeMarks = new int[5];
    for(int j=0; j<5; j++){
        seeMarks[j] = scanner.nextInt();
    }
    student[i].setSEEMarks(seeMarks);
```




```
package CIE;

public class Internals {
    public int[] internalMarks = new int[5];


    public void setInternalMarks(int[] marks) {
        for (int i = 0; i < 5; i++) {
            internalMarks[i] = marks[i];
        }
    }


    public void displayInternalMarks() {
        System.out.print("Internal Marks: ");
        for (int mark : internalMarks) {
            System.out.print(mark + " ");
        }
        System.out.println();
```

```
    }
}
```

```
package CIE;

public class Personal {
    public String usn;
    public String name;
    public int sem;


    public Personal(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }

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

```
package SEE;

import CIE.Personal;

public class External extends Personal {
    public int[] seeMarks = new int[5];


    public External(String usn, String name, int sem) {
        super(usn, name, sem);
    }


    public void setSEEMarks(int[] marks) {
        for (int i = 0; i < 5; i++) {
            seeMarks[i] = marks[i];
        }
    }

    public void displaySEEMarks() {
        System.out.print("SEE Marks: ");
        for (int mark : seeMarks) {
            System.out.print(mark + " ");
        }
        System.out.println();
    }
}
```

```
import CIE.*;
import SEE.*;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
```

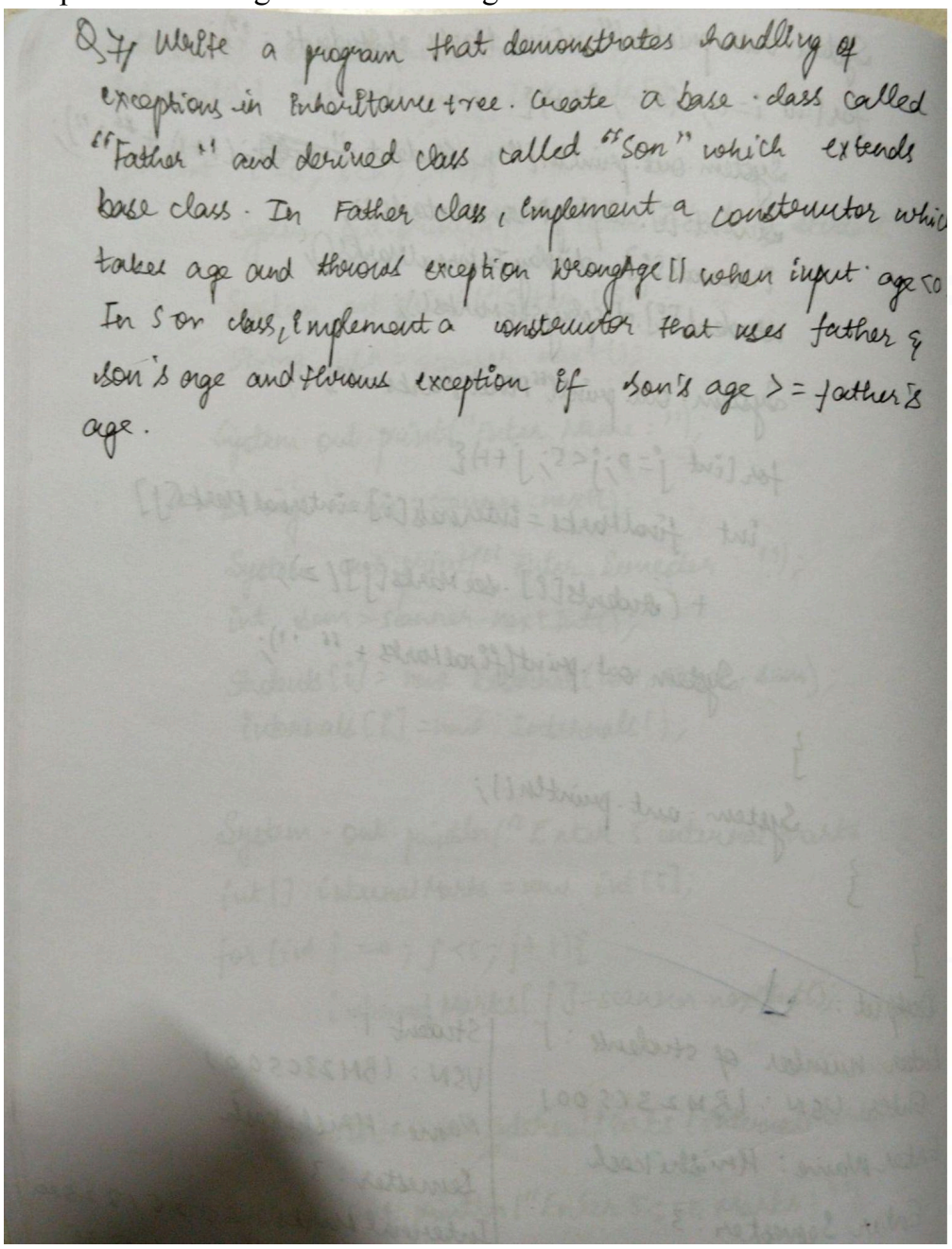
```
        System.out.print("Enter number of students: ");
        int n = scanner.nextInt();

        // Create arrays for objects
        External[] students = new External[n];
        Internals[] internals = new Internals[n];

        // Input details for each student
        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for student " + (i + 1) + ":");

            // Input personal details
            System.out.print("Enter USN: ");
            String usn = scanner.next();
            System.out.print("Enter Name: ");
            String name = scanner.next();
            System.out.print("Enter Semester: ");
            int sem = scanner.nextInt();

            // Create objects
            students[i] = new External(usn, name, sem);
            internals[i] = new Internals();

            // Input internal marks
            System.out.println("Enter 5 internal marks: ");
            int[] internalMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                internalMarks[j] = scanner.nextInt();
            }
            internals[i].setInternalMarks(internalMarks);

            // Input SEE marks
            System.out.println("Enter 5 SEE marks: ");
            int[] seeMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                seeMarks[j] = scanner.nextInt();
            }
            students[i].setSEEMarks(seeMarks);
        }

        // Display details and calculate final marks
        System.out.println("\nFinal Marks of Students:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nStudent " + (i + 1) + ":");
            students[i].displayPersonalDetails();
            internals[i].displayInternalMarks();
            students[i].displaySEEMarks();

            System.out.print("Final Marks: ");
            for (int j = 0; j < 5; j++) {
                int finalMarks = internals[i].internalMarks[j] + (students[i].seeMarks[j] / 2);
                System.out.print(finalMarks + " ");
            }
            System.out.println();
        }

        scanner.close();
    }
}
```

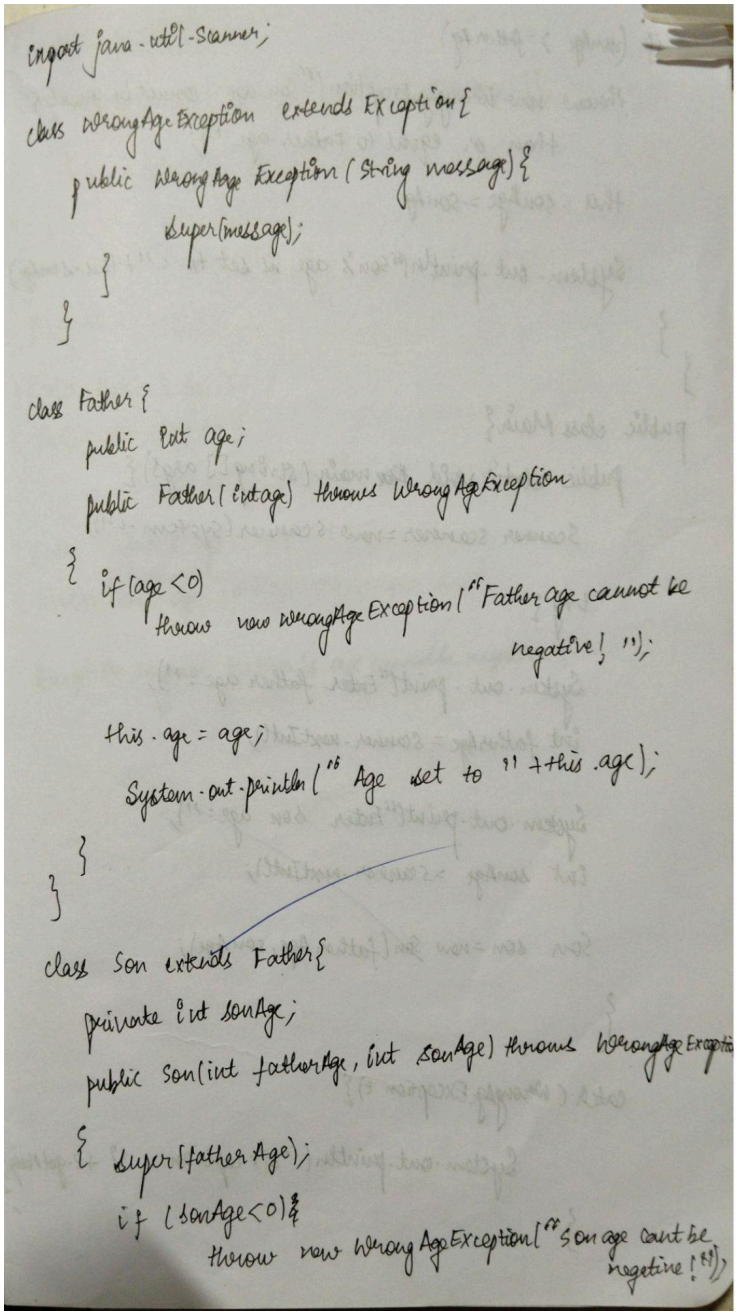
OUTPUT

```
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week5> cd "c:\Users\HP\Desktop\coding\
va Main }
Enter number of students: 1

Enter details for student 1:
Enter USN: 1BM21CS001
Enter Name: Alice
Enter Semester: 3
Enter 5 internal marks:
20 18 19 22 21
Enter 5 SEE marks:
60 70 65 80 75

Final Marks of Students:

Student 1:
USN: 1BM21CS001
Name: Alice
Semester: 3
Internal Marks: 20 18 19 22 21
SEE Marks: 60 70 65 80 75
Final Marks: 50 53 51 62 58
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week6>
```

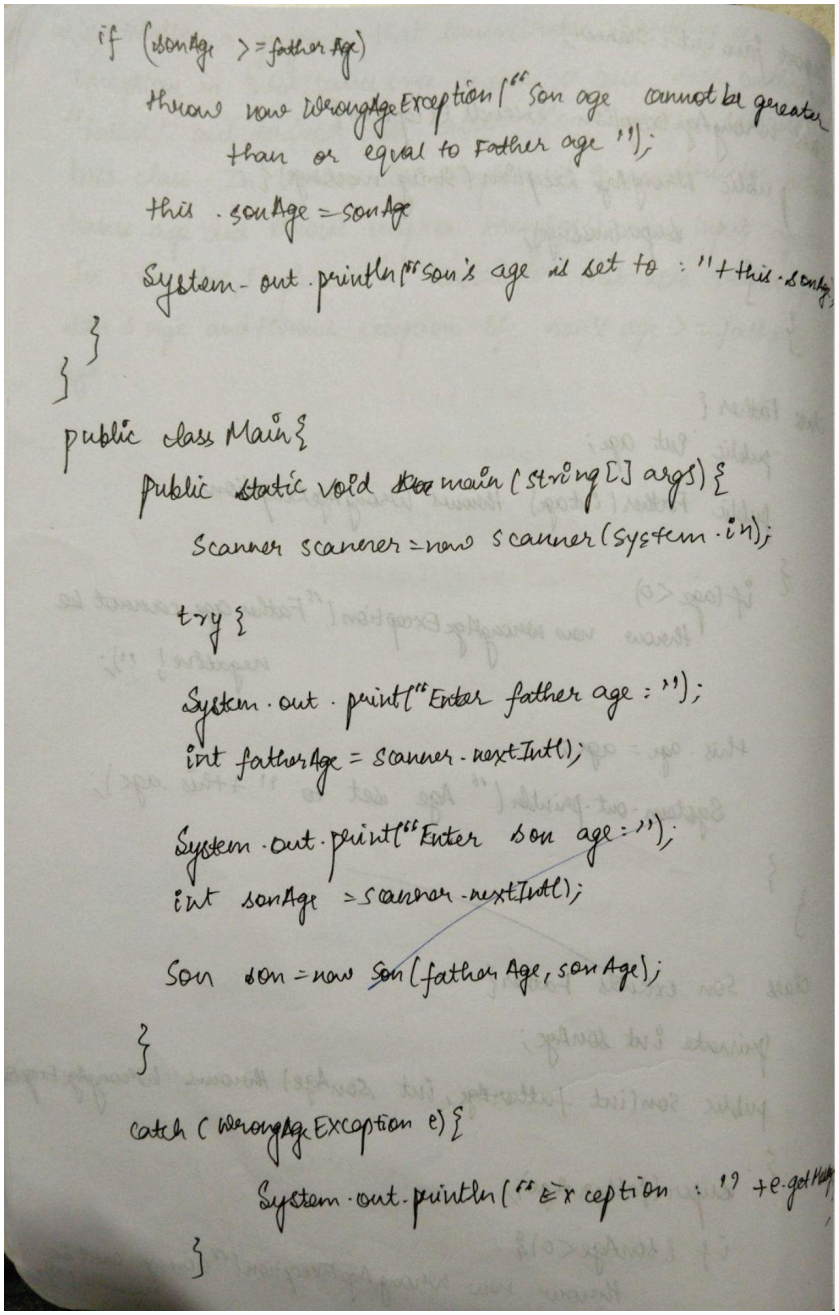
LABORATORY PROGRAM - 7

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<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.

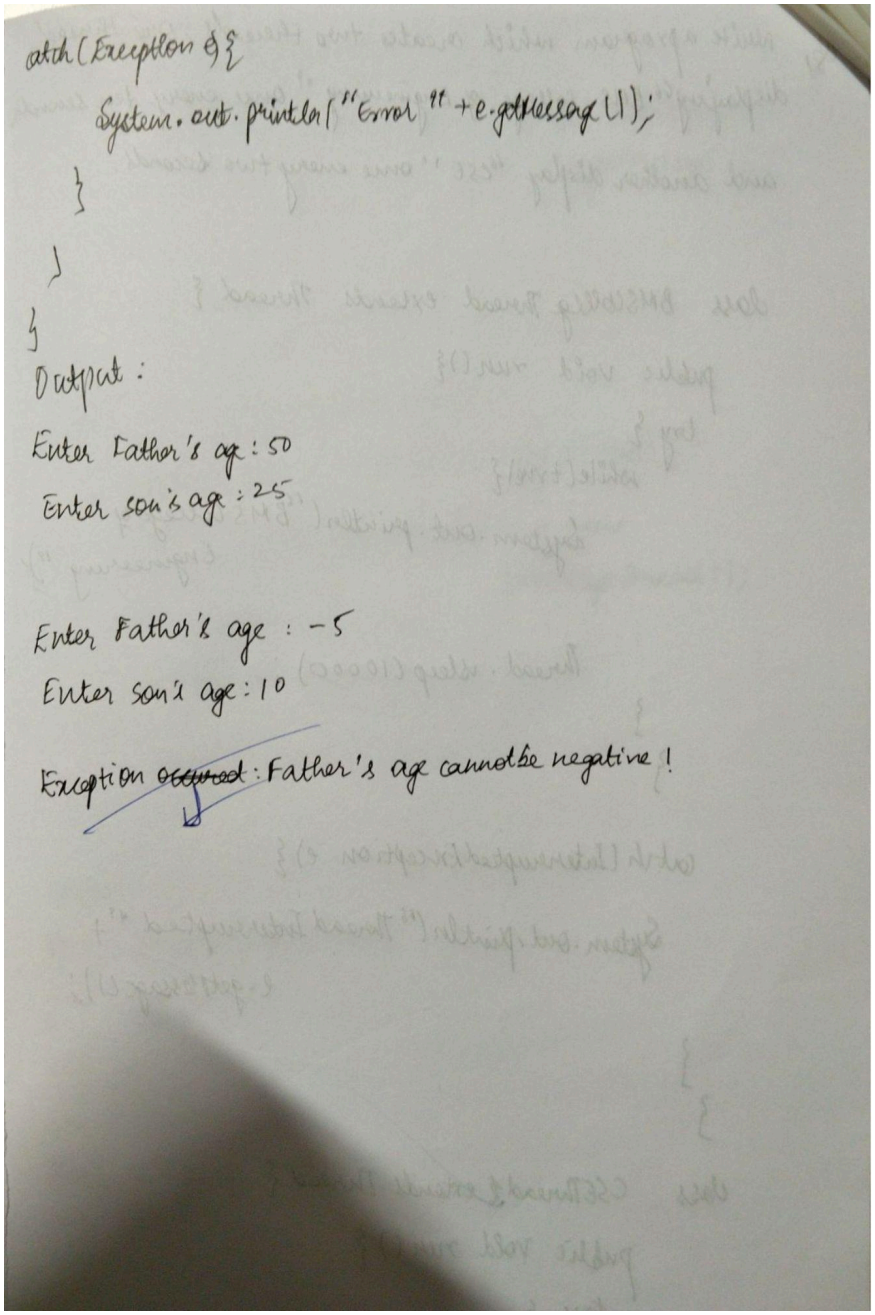




```
import java.util.Scanner;

class WrongAgeException extends Exception{
    public WrongAgeException(String message){
        super(message);
    }
}

class Father{
    public int age;
    public Father(int age) throws WrongAgeException
    {
        if(age<0)
            throw new WrongAgeException("Father age cannot be
                                         negative!");

        this.age = age;
        System.out.println("Age set to "+this.age);
    }
}

class Son extends Father{
    private int sonAge;
    public Son(int fatherAge, int sonAge) throws WrongAgeExcepti

    {   super(fatherAge);
        if(sonAge<0){
            throw new WrongAgeException("Son age cant be
                                         negetive!");
```



```
if (sonAge >= fatherAge)
    throw new WrongAgeException("Son age cannot be greater
        than or equal to father age");
    this.sonAge = sonAge
    System.out.println("son's age is set to : "+this.sonAge);
}
}

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

        try{
        System.out.print("Enter father age : ");
        int fatherAge = scanner.nextInt();

        System.out.print("Enter son age : ");
        int sonAge = scanner.nextInt();

        Son son = new Son(fatherAge, sonAge);
    }
    catch (WrongAgeException e){
            System.out.println("Exception : "+e.getMes
    }
```




```
import java.util.Scanner;

class WrongAgeException extends Exception {
    public WrongAgeException(String message) {
        super(message);
    }
}


class Father {
    protected int age;


    public Father(int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Father's age cannot be negative!");
        }
        this.age = age;
        System.out.println("Father's age is set to: " + this.age);
```

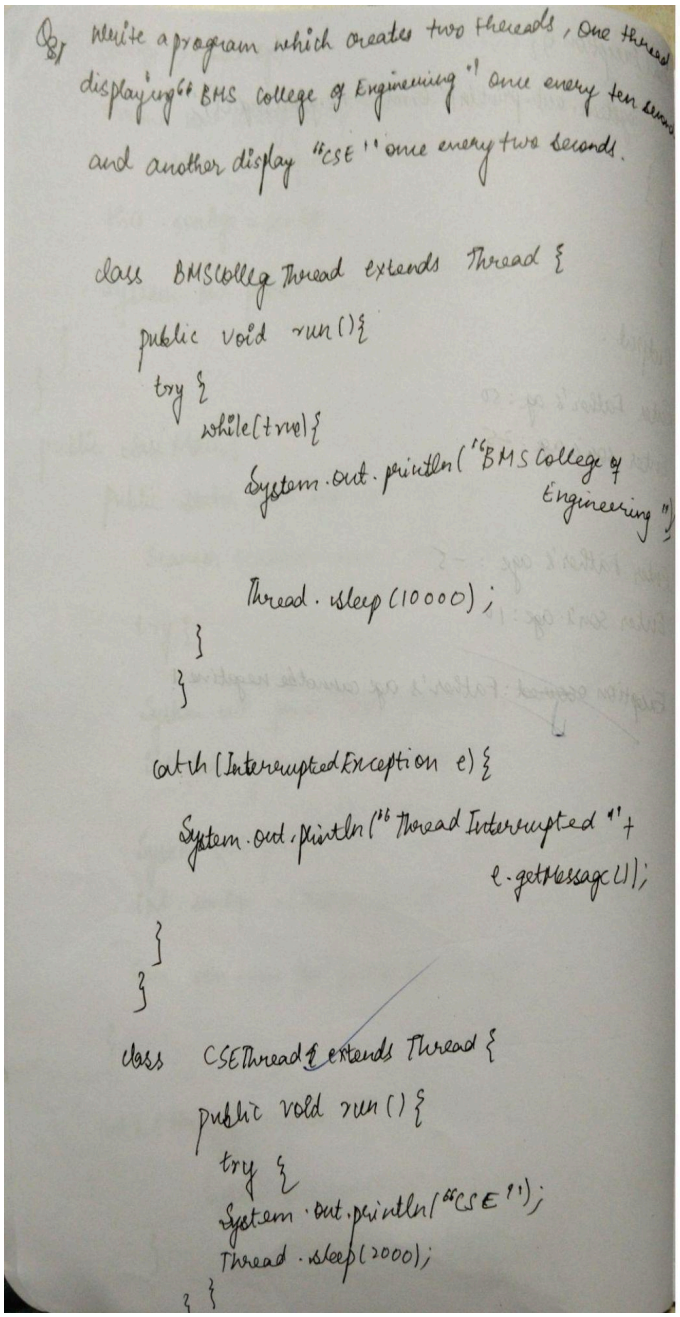
```
    }
}

class Son extends Father {
    private 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 is set to: " + this.sonAge);
    }
}


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

        try {
            System.out.print("Enter Father's age: ");
            int fatherAge = scanner.nextInt();


            System.out.print("Enter Son's age: ");
            int sonAge = scanner.nextInt();


            Son son = new Son(fatherAge, sonAge);
        } catch (WrongAgeException e) {

            System.out.println("Exception occurred: " + e.getMessage());
        } catch (Exception e) {
            System.out.println("An unexpected error occurred: " + e.getMessage());
        } finally {
            scanner.close();
            System.out.println("Program execution completed.");
        }
    }
}
```

OUTPUT

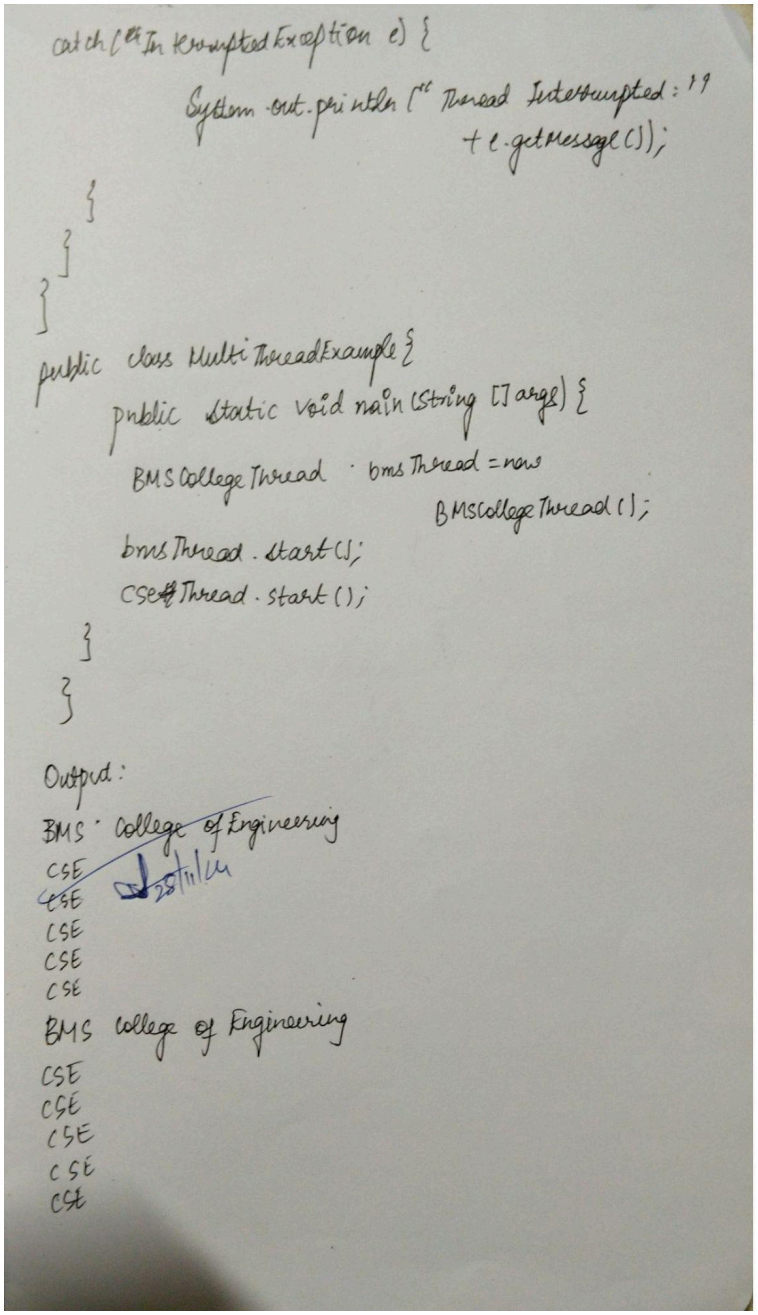
```
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week6> cd "c:\Users
va Main }
Enter Father's age: 50
Enter Son's age: 25
Father's age is set to: 50
Son's age is set to: 25
Program execution completed.
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week7> cd "c:\Users
va Main }
Enter Father's age: -5
Enter Son's age: 10
Exception occurred: Father's age cannot be negative!
Program execution completed.
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week7>
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week7>
```

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.



```
Q8) Write a program which creates two threads, One thread
displaying "BMS College of Engineering" once every ten seconds
and another display "CSE" once every two seconds.

class BMSCollegThread extends Thread {
  public void run(){
    try {
      while(true){
        System.out.println("BMS College of
                                Engineering");
        Thread.sleep(10000);
      }
    }
    catch (InterruptedException e){
      System.out.println("Thread Interrupted"+
                          e.getMessage());
    }
}
class CSEThread extends Thread {
    public void run(){
      try {
      System.out.println("CSE");
      Thread.sleep(2000);
    }}
```




```
class BMSCollegeThread extends Thread{
   public void run(){
      try {
         for(int i=0;i<2;i++){
            System.out.println("BMS College of Engineering");
            Thread.sleep(10000);
         }

      } catch (InterruptedException e) {
         System.out.println("Thread Interrupted "+e.getMessage());
      }
   }
}
class CSEThread extends Thread{
   public void run(){
      try {
            for(int i=0;i<10;i++){
            System.out.println("CSE");
```

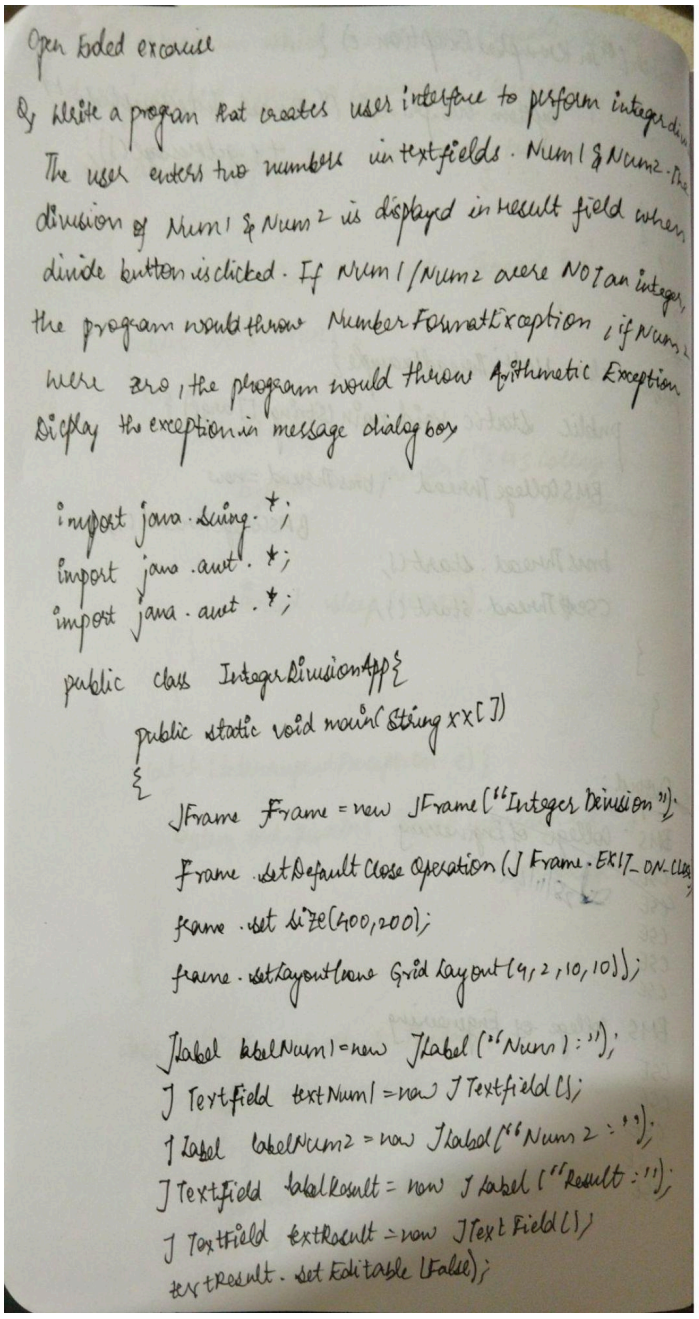
```
            Thread.sleep(2000);
            }
        }

    catch (InterruptedException e) {
        System.out.println("Thread Interrupted "+e.getMessage());
    }
  }
}
public class MultiThreadExample {
  public static void main(String[] args) {
    BMSCollegeThread bms = new BMSCollegeThread();
    CSEThread cse = new CSEThread();
    bms.start();
    cse.start();
  }


}
```

OUTPUT

```
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava> cd "c:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week8\" ; if ($?) { javac MultiThreadExample.java } ; if ($?) { java MultiThreadExample }
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
PS C:\Users\HP\Desktop\coding\1BM23CS367\OOPwithJava\Week8>
```

LABORATORY PROGRAM - 9

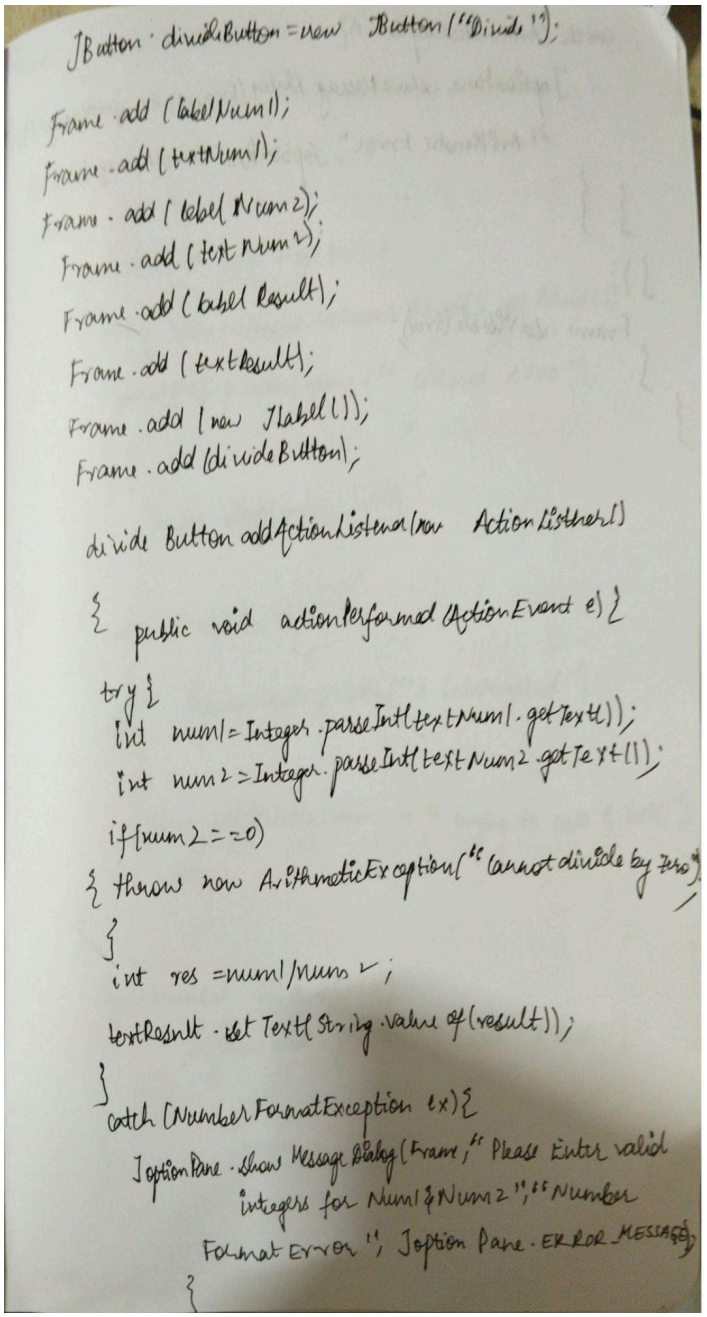
Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException.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



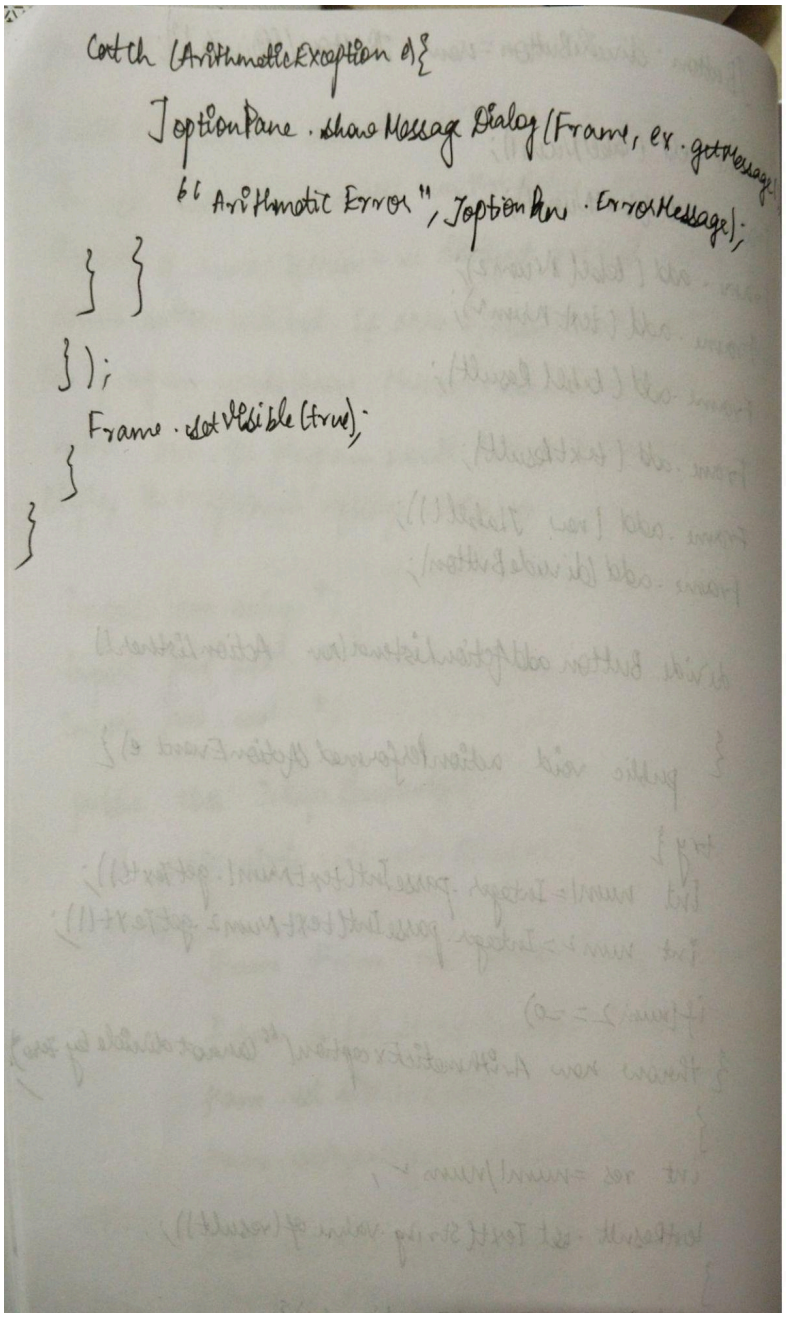
Open ended exercise

Q: Write a program that creates user interface to perform integer division. The user enters two numbers in textfields. Num1 & Num2. The division of Num1 & Num2 is displayed in Result field when divide button is clicked. If Num1/Num2 were NOT an integer, the program would throw NumberFormatException, if Num2 were zero, the program would throw Arithmetic Exception. Display the exception in message dialog box.

```
import java.lang.*;
import java.awt.*;
import java.awt.*;

public class IntegerDivisionApp{
    public static void main(String xx[])
    {
        JFrame Frame = new JFrame("Integer Division");
        Frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400,200);
        frame.setLayout(new GridLayout(4,2,10,10));

        JLabel labelNum1=new JLabel("Num1:");
        JTextfield textNum1=new JTextfield();
        JLabel labelNum2=new JLabel("Num2:");
        JTextfield labelResult=new JLabel("Result:");
        JTextfield txtResult=new JTextField();
        textResult.setEditable(false);
```



```
JButton divideButton = new JButton("Divide");

Frame.add(labelNum1);
Frame.add(textNum1);
Frame.add(labelNum2);
Frame.add(textNum2);
Frame.add(labelResult);
Frame.add(textResult);
Frame.add(new JLabel());
Frame.add(divideButton);

divideButton.addActionListener(new ActionListener()
{ public void actionPerformed(ActionEvent e){
try{
int num1 = Integer.parseInt(textNum1.getText());
int num2 = Integer.parseInt(textNum2.getText());
if(num2==0)
{ throw new ArithmeticException("Cannot divide by zero");
}
int res = num1/num2;
textResult.setText(String.valueOf(result));
}
catch(NumberFormatException ex){
JOptionPane.showMessageDialog(Frame," Please Enter valid
integers for Num1 & Num2","Number
Format Error", JOptionPane.ERROR_MESSAGE);
}
```




```
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(number
        1);
        add(num1);
        add(number
        2);
        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);
```

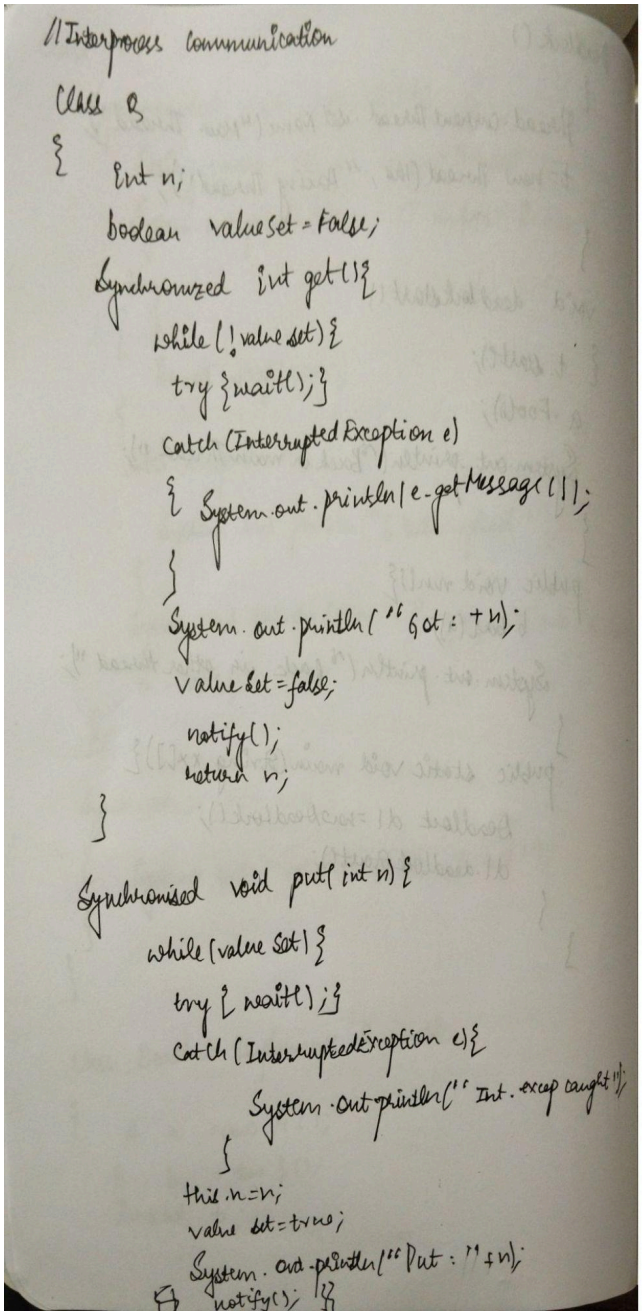
```
            }
        });
}
public void actionPerformed(ActionEvent ae)
{
        int
        n1,n2;
        try
        {
                if (ae.getSource() == dResult)
                {
                        n1=Integer.parseInt(num1.getText());
                        n2=Integer.parseInt(num2.getText());

                        /*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(),outR
```

```
esult.getY()+outResult. getHeight()-8);
else
g.drawString(out,1
00,200); flag=0;
    }
```

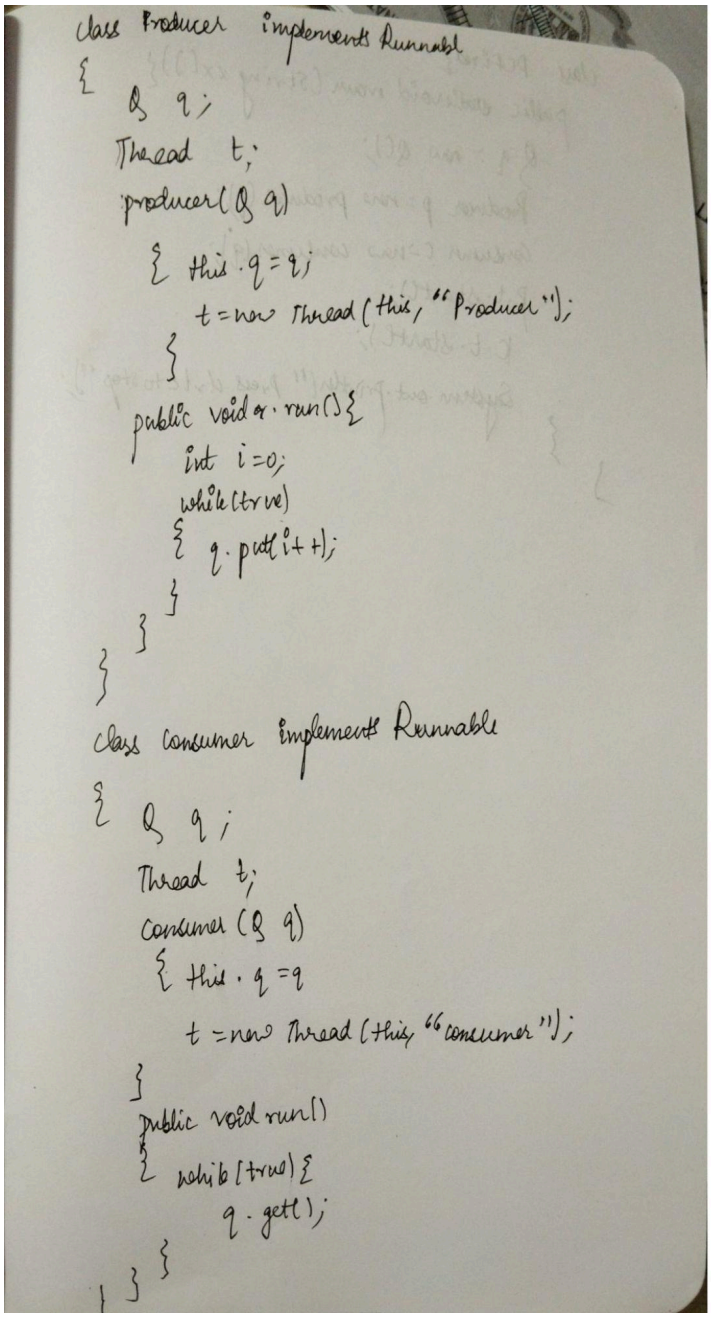
```
Put: 23034
Got: 23034
Put: 23035
Got: 23035
Put: 23036
Got: 23036
Put: 23037
Got: 23037
Put: 23038
Got: 23038
Put: 23039
Got: 23039
Put: 23040
Got: 23040
Put: 23041
Got: 23041
Put: 23042
Got: 23042
Put: 23043
Got: 23043
Put: 23044
Got: 23044
Put: 23045
Got: 23045
Put: 23046
Got: 23046
Put: 23047
Got: 23047
Put: 23048
Got: 23048
Put: 23049
Got: 23049
Put: 23050
Got: 23050
```

LABORATORY PROGRAM - 10

Demonstrate Interprocess communication and deadlock



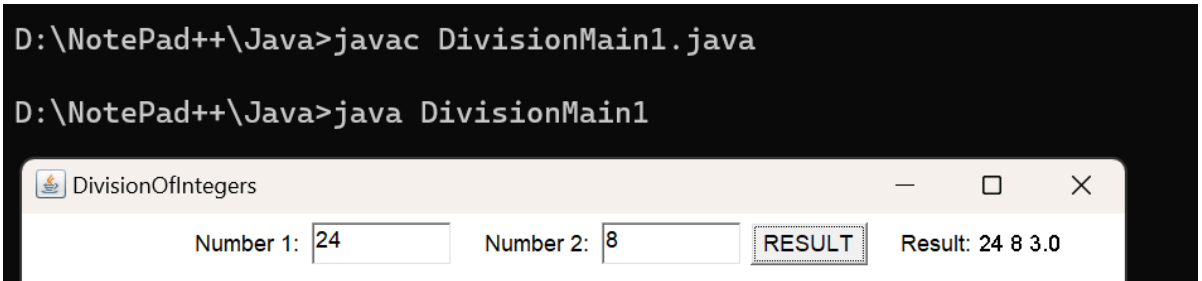
```
//Interprocess communication
Class Q
{
    int n;
    boolean valueSet = false;
    Synchronized int get(){
        while(!valueSet){
        try {wait();}
        catch (InterruptedException e)
        { System.out.println(e.getMessage());
        }
        System.out.println("Got : "+n);
        valueSet=false;
        notify();
        return n;
    }
    Synchronised void put(int n){
        while(valueSet){
        try {wait();}
        Catch(InterruptedException e){
            System.out.println("Int. excep caught");
        }
        this.n=n;
        valueSet=true;
        System.out.println("Put : "+n);
        notify();
```



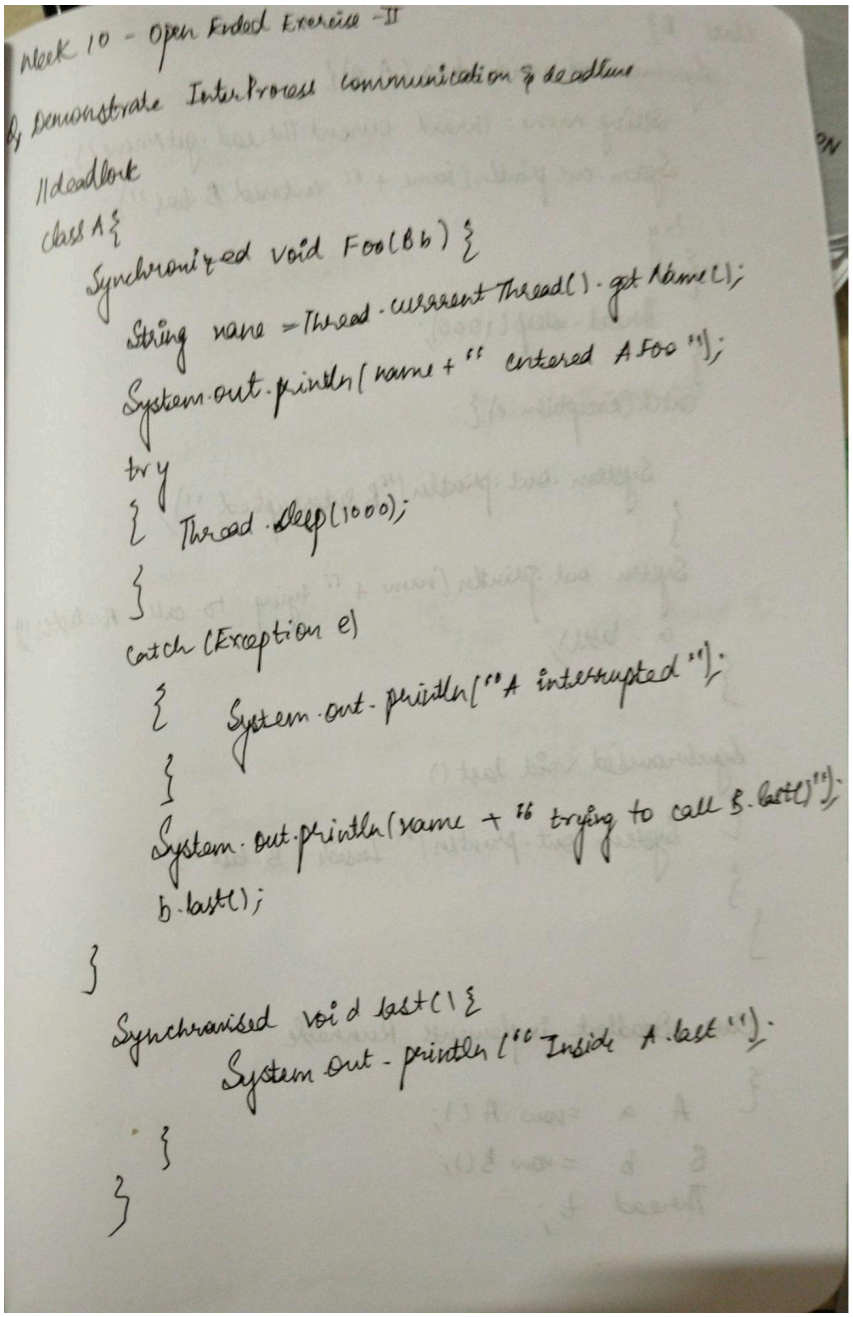

```
class Q {
int n;
boolean valueSet = false;

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.");
}
}
```

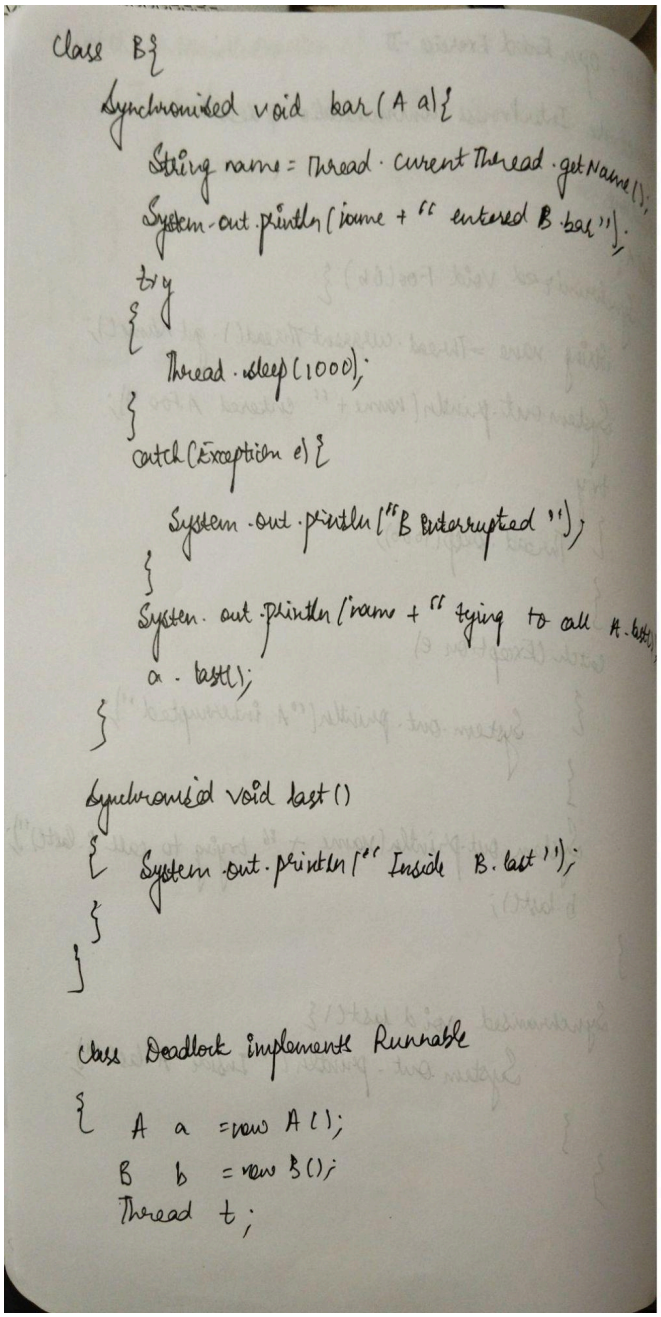
OUTPUT



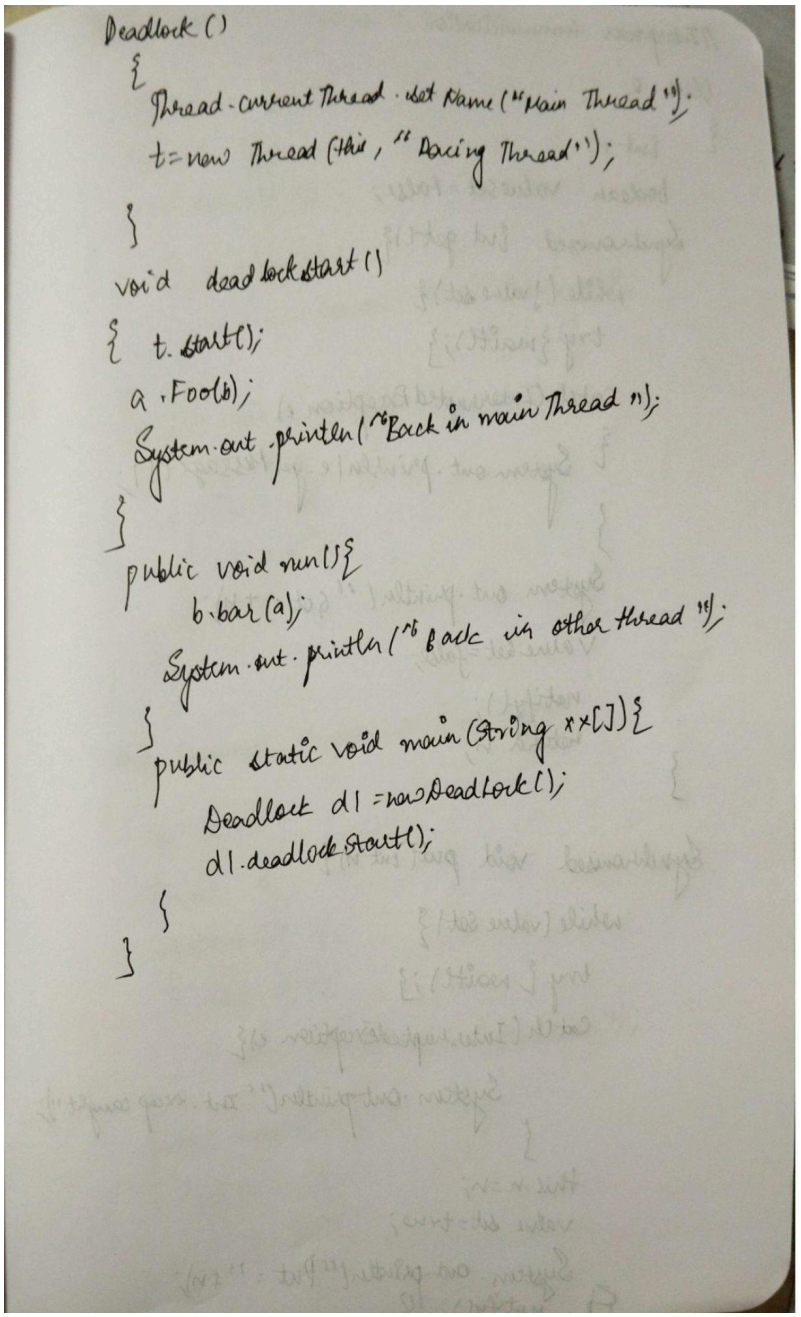
ii. Demonstration of deadlock



```
Week 10 - Open Ended Exercise -II
By demonstrate Inter Process communication & deadlock
//deadlock
class A{
  Synchronized void Foo(B b){
    String name = Thread.currentThread().getName();
    System.out.println(name + " entered A.foo");
    try
    { Thread.sleep(1000);
    }
    catch (Exception e)
    { System.out.println("A interrupted");
    }
    System.out.println(name + " trying to call B.last()");
    b.last();
  }
  Synchronized void last(){
      System.out.println("Inside A.last");
  }
}
```

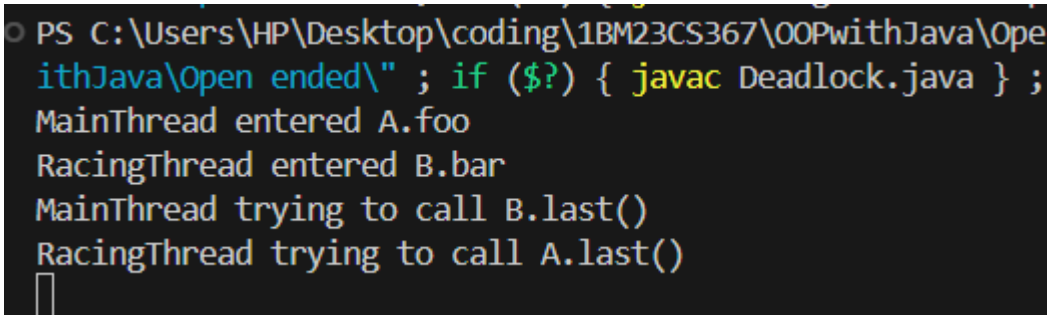


```
class B{
    synchronized void bar(A a){
        String name = Thread.currentThread.getName();
        System.out.println(name + " entered B.bar");
        try
        {
            Thread.sleep(1000);
        }
        catch(Exception e){
            System.out.println("B interrupted");
        }
        System.out.println(name + " trying to call A.last()");
        a.last();
    }

    synchronized void last()
    {   System.out.println(" Inside B.last");
    }
}

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




```
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(); }
}
```



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