VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

SAAHYA K S (1BM23CS368)

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 **SAAHYA K S(1BM23CS368),** who is a 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

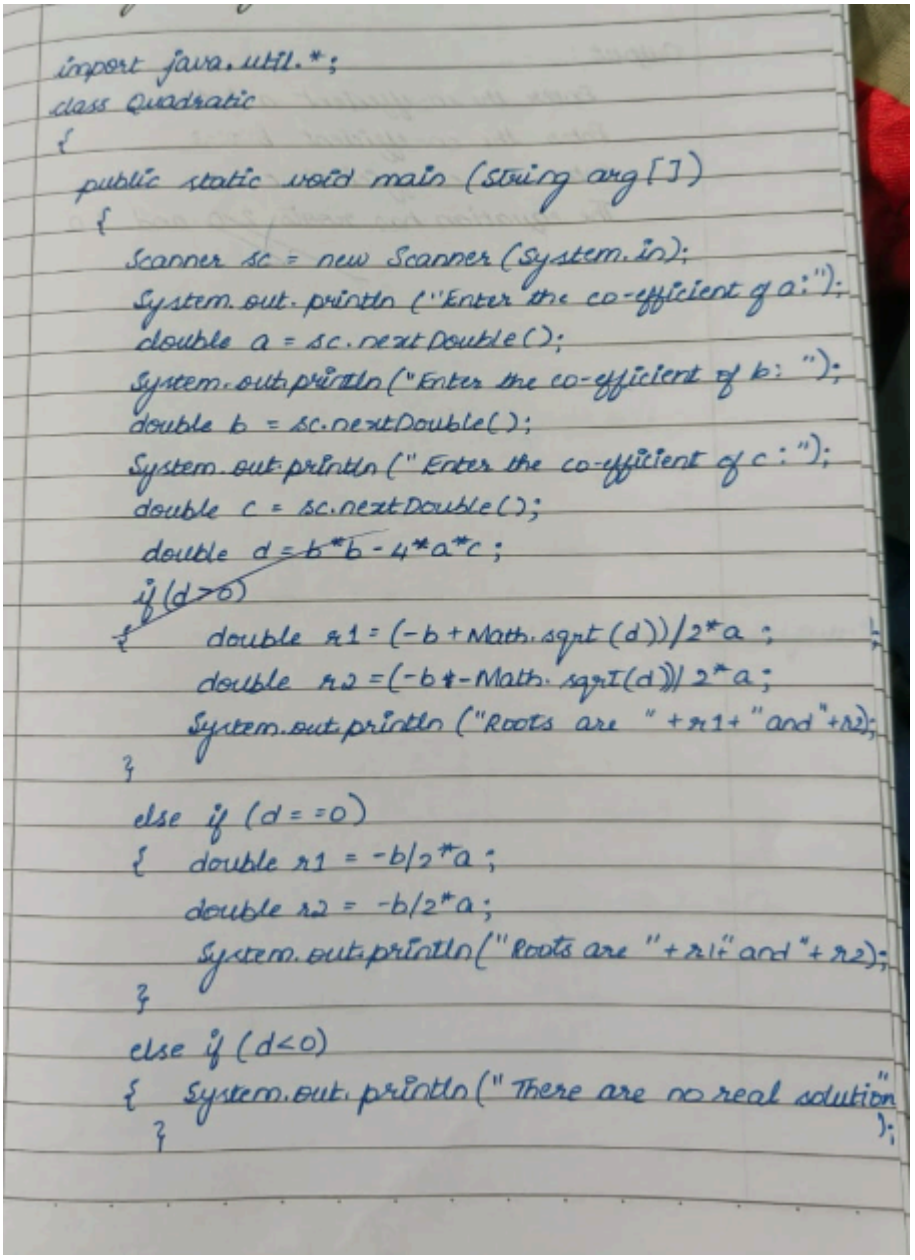
Professor and Head,
Department of CSE
BMSCE, Bengaluru

INDEX

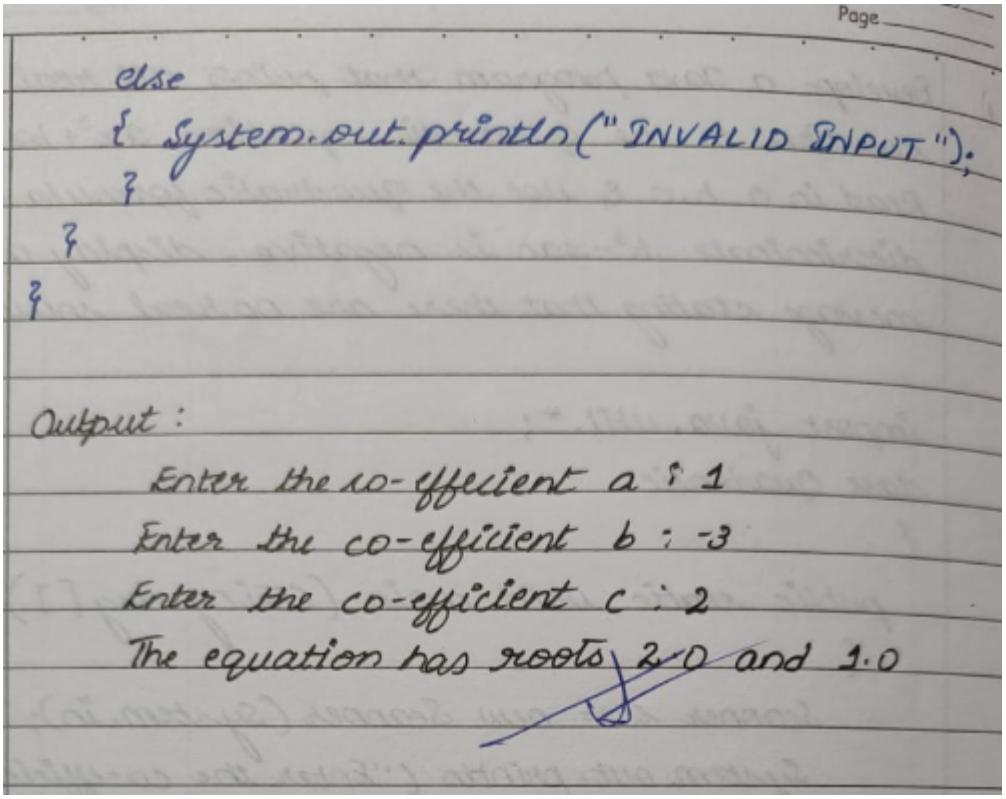
LABORATORY PROGRAM - 1

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.

Algorithm:



```
import java.util.*;
class Quadratic
{
 public static void main (String arg[])
  {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter the co-efficient of a:");
    double a = sc.nextDouble();
    System.out.println("Enter the co-efficient of b: ");
    double b = sc.nextDouble();
    System.out.println("Enter the co-efficient of c:");
    double c = sc.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("Roots are " +r1+ "and "+r2);
    }
    else if (d==0)
    { double r1 = -b/2*a;
      double r2 = -b/2*a;
      System.out.println("Roots are "+r1+"and "+r2);
    }
    else if (d<0)
    { System.out.println("There are no real solution");
    }
```



Code:

```
import java.util.Scanner;
class Quadratic
{
  public static void main(String arg[]){
   Scanner input= new Scanner(System.in);
   System.out.println("Enter co-efficients of a:");
   double a =input.nextDouble();
   System.out.println("Enter co-efficients of b:");
   double b=input.nextDouble();
   System.out.println("Enter co-efficients of c:");
   double c=input.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("Roots are : "+r1+" "+r2);
    }
    else if(d==0){
        double r1=-b/2*a;
        double r2=-b/2*a;
        System.out.println("Roots are : "+r1+" "+r2);
    }
    else if(d<0){
        System.out.println("Roots are imaginary ");
    }
    else{
        System.out.println("Invalid input");
    }
   }
}
```

Output:

```
Enter co-efficients of a:
1
Enter co-efficients of b:
9
Enter co-efficients of c:
5
Roots are : -0.594875162046673 -8.405124837953327
```

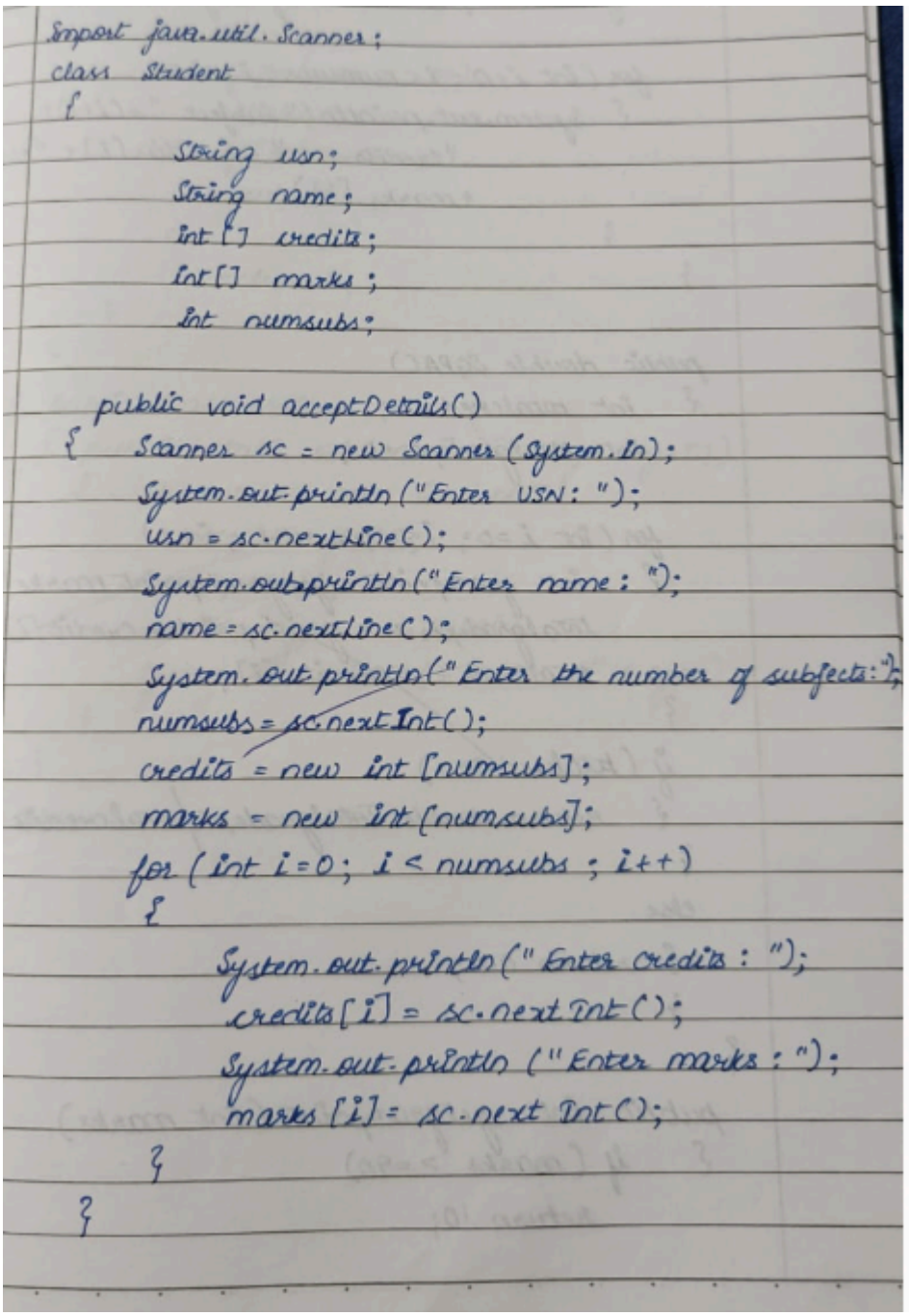
```
Enter co-efficients of a:
5
Enter co-efficients of b:
7
Enter co-efficients of c:
2
Roots are : -10.0 -25.0
```

```
Enter co-efficients of a:
1
Enter co-efficients of b:
2
Enter co-efficients of c:
7
Roots are imaginary
```

LABORATORY PROGRAM - 2

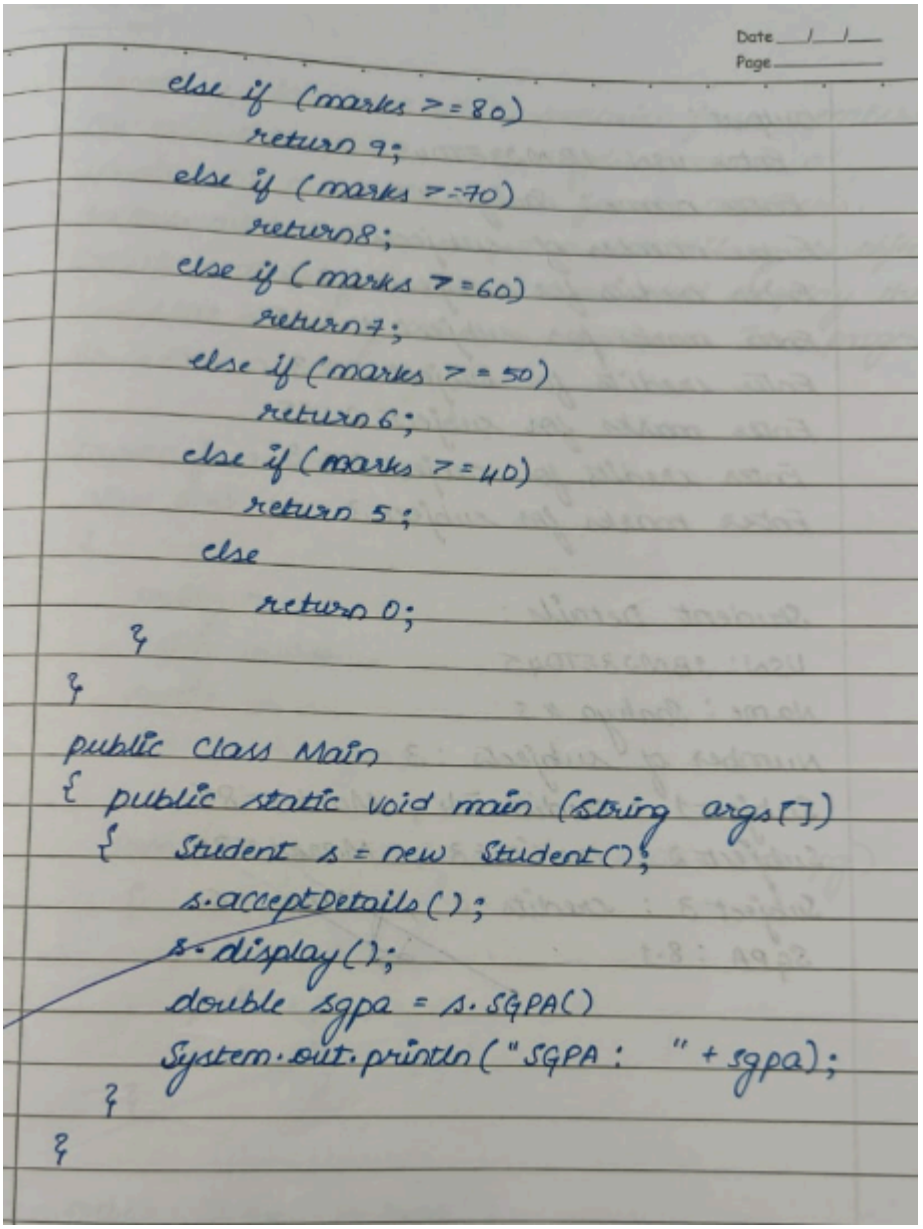
Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

Algorithm:



```
import java.util.Scanner;
class Student
{
    String usn;
    String name;
    int[] credits;
    int[] marks;
    int numsubs;

  public void acceptDetails()
{   Scanner sc = new Scanner(System.in);
    System.out.println("Enter USN: ");
    usn = sc.nextLine();
    System.out.println("Enter name: ");
    name = sc.nextLine();
    System.out.println("Enter the number of subjects:");
    numsubs = sc.nextInt();
    credits = new int[numsubs];
    marks = new int[numsubs];
    for(int i=0; i<numsubs; i++)
    {
        System.out.println("Enter credits: ");
        credits[i] = sc.nextInt();
        System.out.println("Enter marks: ");
        marks[i] = sc.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 totalcred = 0;
   int totalgradepts =0;

   for(int i=0; i<numsubs ; i++)
   {  int gradepoint = getgradepoint marks[i];
      totalgradepts + = gradepoint + Credit[i];
      totalcred + = credits[i];
   }
   if(totalcred >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[])
  { Student s = new Student();
    s.acceptDetails();
    s.display();
    double sgpa = s.SGPA()
    System.out.println("SGPA :  " + sgpa);
  }
}
```

Code:

```
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);
    }
}
```

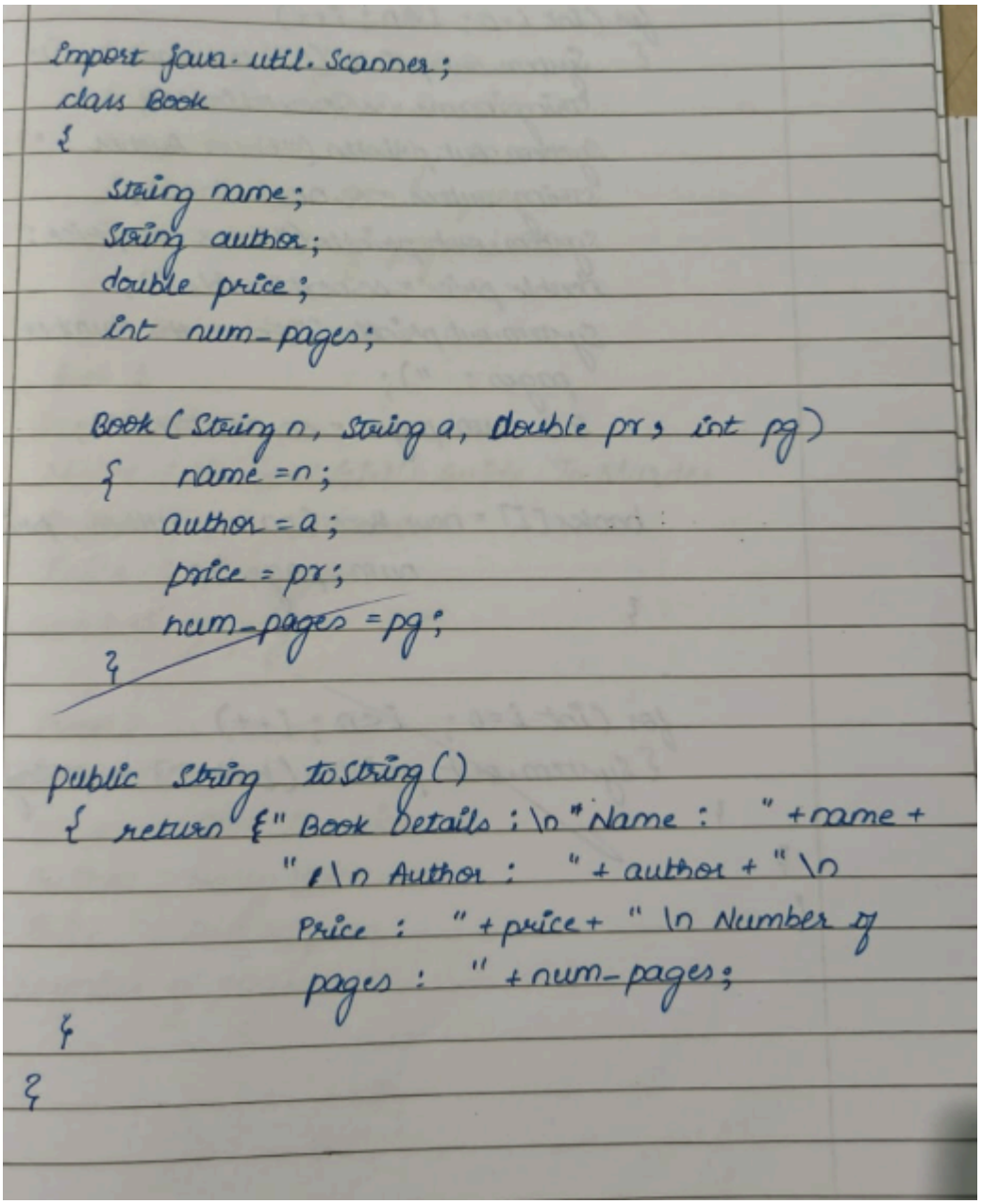
Output:

```
Enter USN: 1BM23CS368
Enter Name: Saahya K S
Enter the number of subjects: 5
Enter credits for subject 1: 4
Enter marks for subject 1: 95
Enter credits for subject 2: 3
Enter marks for subject 2: 87
Enter credits for subject 3: 3
Enter marks for subject 3: 89
Enter credits for subject 4: 2
Enter marks for subject 4: 90
Enter credits for subject 5: 1
Enter marks for subject 5: 93

Student Details:
USN: 1BM23CS368
Name: Saahya K S
Number of Subjects: 5
Subject 1: Credits = 4, Marks = 95
Subject 2: Credits = 3, Marks = 87
Subject 3: Credits = 3, Marks = 89
Subject 4: Credits = 2, Marks = 90
Subject 5: Credits = 1, Marks = 93

SGPA: 9.538461538461538
```

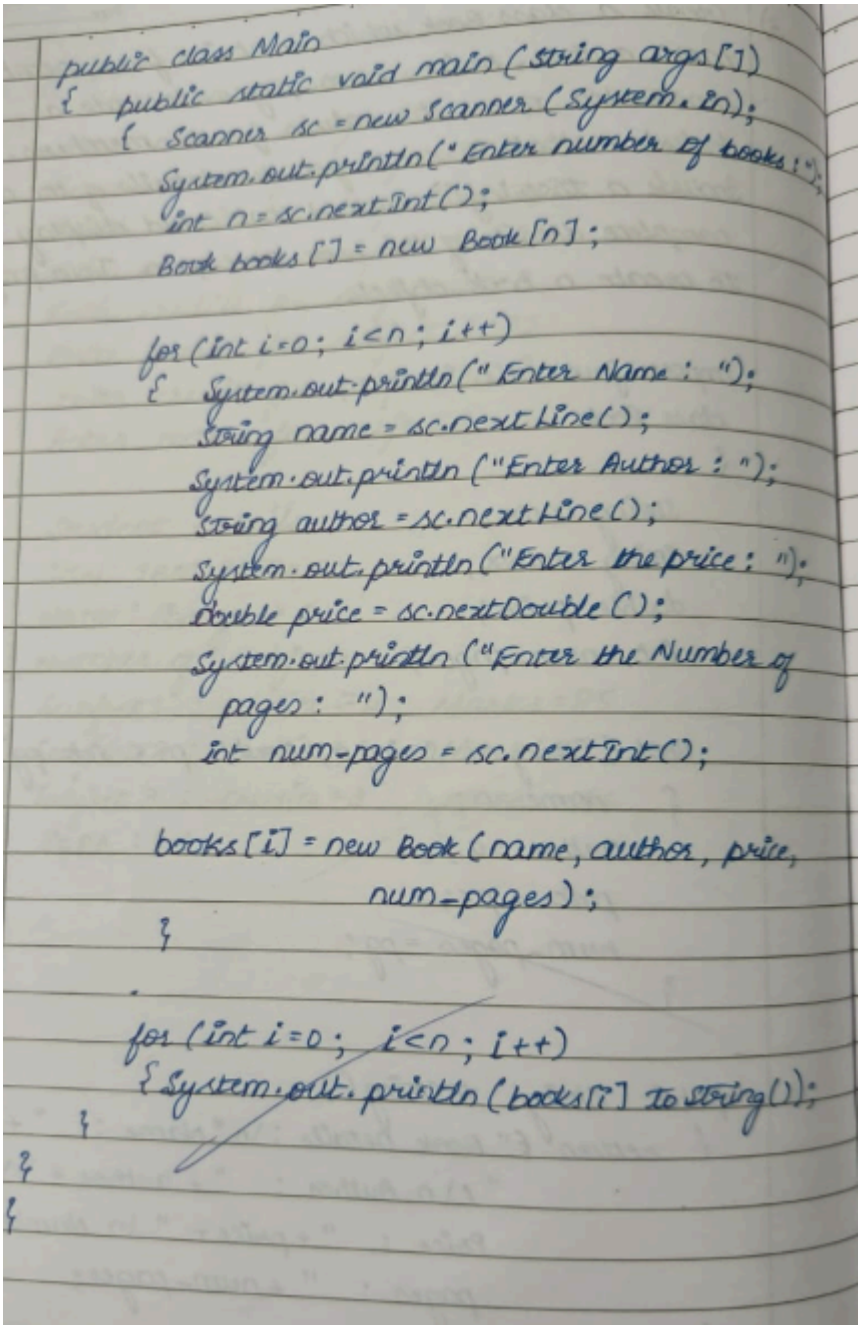
LABORATORY PROGRAM - 3

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.

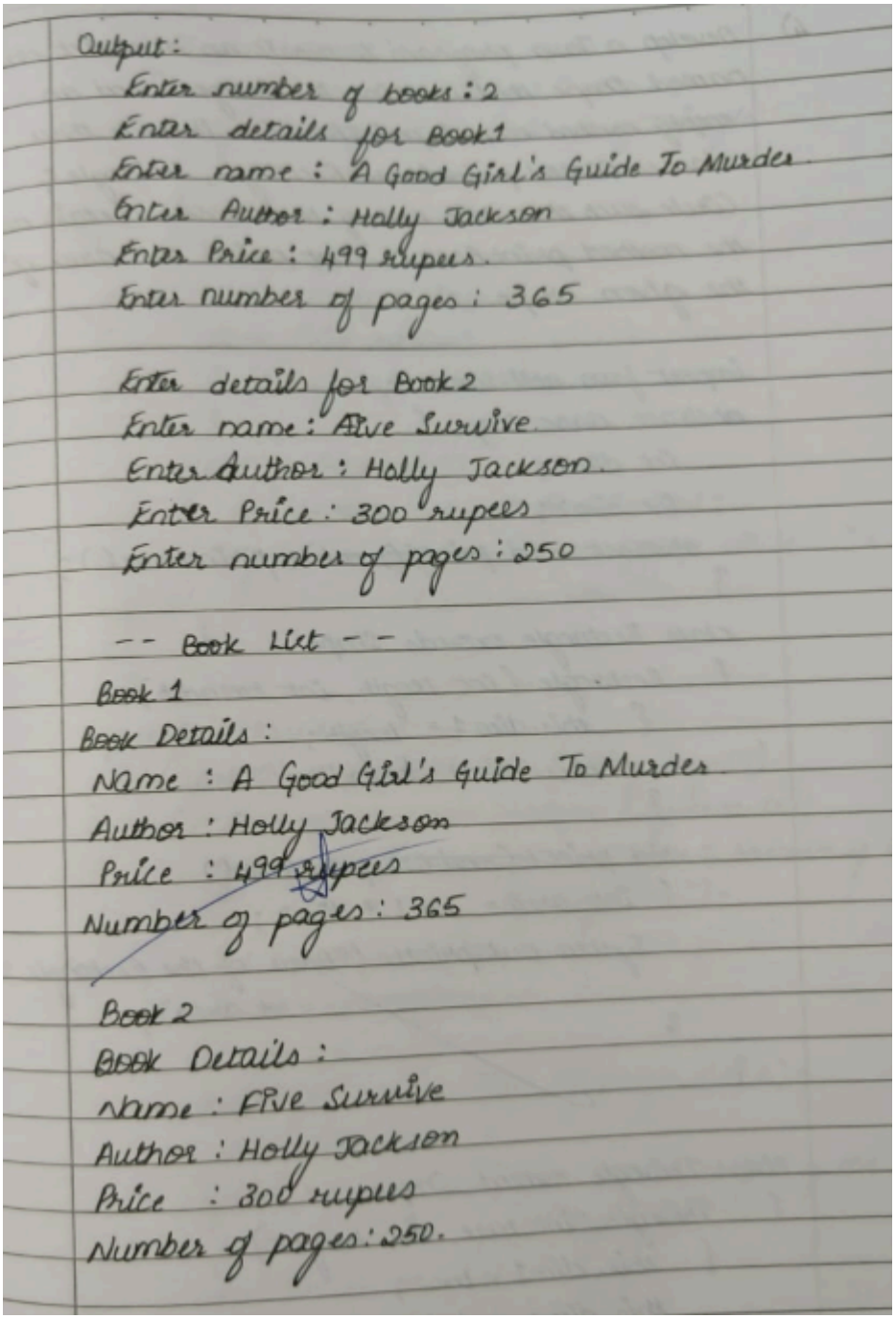
Algorithm:



```
import java.util.Scanner;
class Book
{
    String name;
    String author;
    double price;
    int num_pages;

    Book(String n, String a, double pr, int pg)
    {   name = n;
        author = a;
        price = pr;
        num_pages = pg;
    }

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 sc = new Scanner (System.in);
  System.out.println("Enter number of books:");
  int n = sc.nextInt();
  Book books[] = new Book[n];

  for (int i=0; i<n; i++)
  { System.out.println("Enter Name : ");
    String name = sc.nextLine();
    System.out.println ("Enter Author : ");
    String author = sc.nextLine();
    System.out.println("Enter the price : ");
    Double price = sc.nextDouble();
    System.out.println ("Enter the Number of
      pages : ");
    int num_pages = sc.nextInt();

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

  for (int i=0; i<n; i++)
  {System.out.println (books[i] to String());
  }
}
}
```



Code:

```
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());
        }
    }
}
```

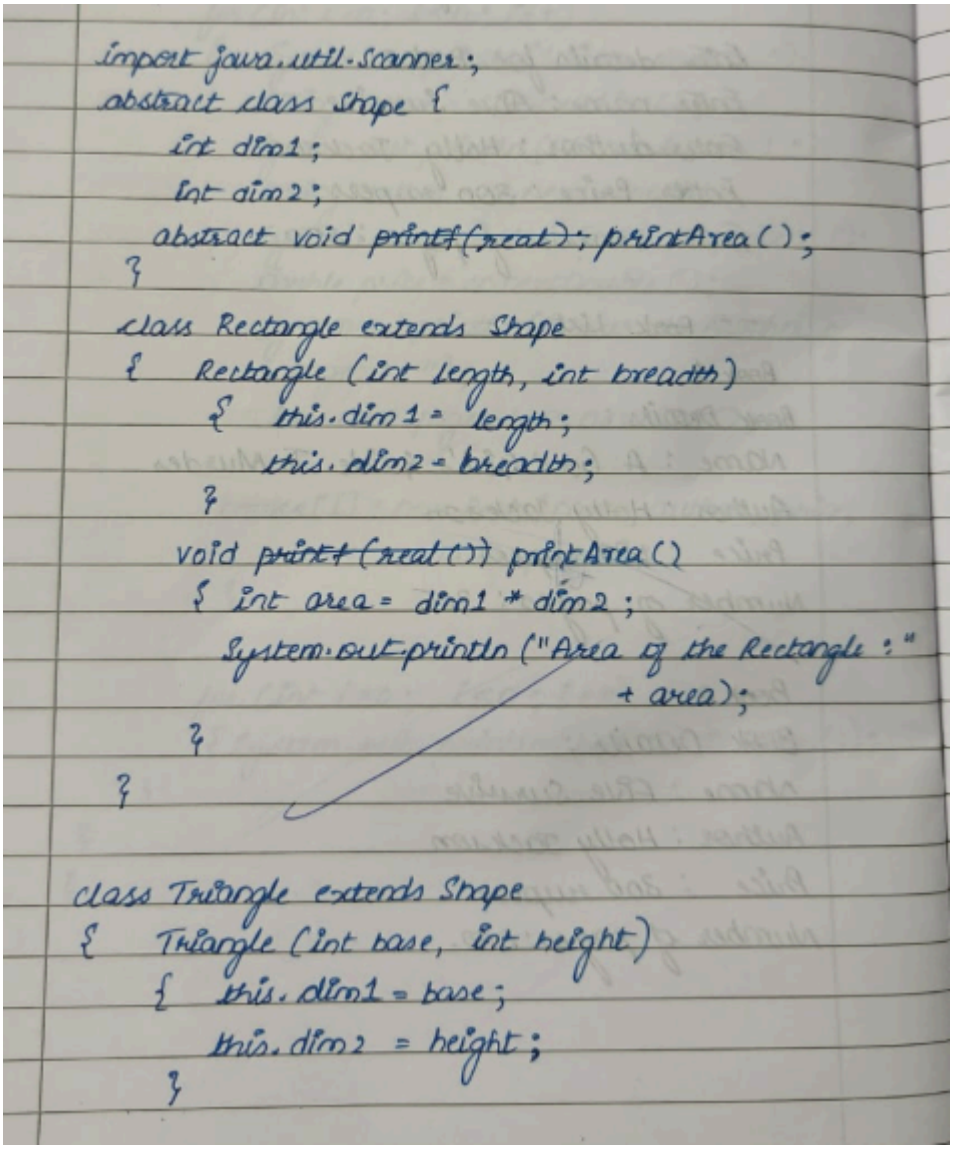
Output:

```
Enter the number of books: 2

Enter details for Book 1:
Enter Name: Five Survive
Enter Author: Holly Jackson
Enter Price: 499
Enter Number of Pages: 550

Enter details for Book 2:
Enter Name: Attack on Titan
Enter Author: Hajime Isayama
Enter Price: 300
Enter Number of Pages: 200

--- Book List ---

Book 1:
Book Details:
Name: Five Survive
Author: Holly Jackson
Price: $499.0
Number of Pages: 550

Book 2:
Book Details:
Name: Attack on Titan
Author: Hajime Isayama
Price: $300.0
Number of Pages: 200
```

LABORATORY PROGRAM - 4

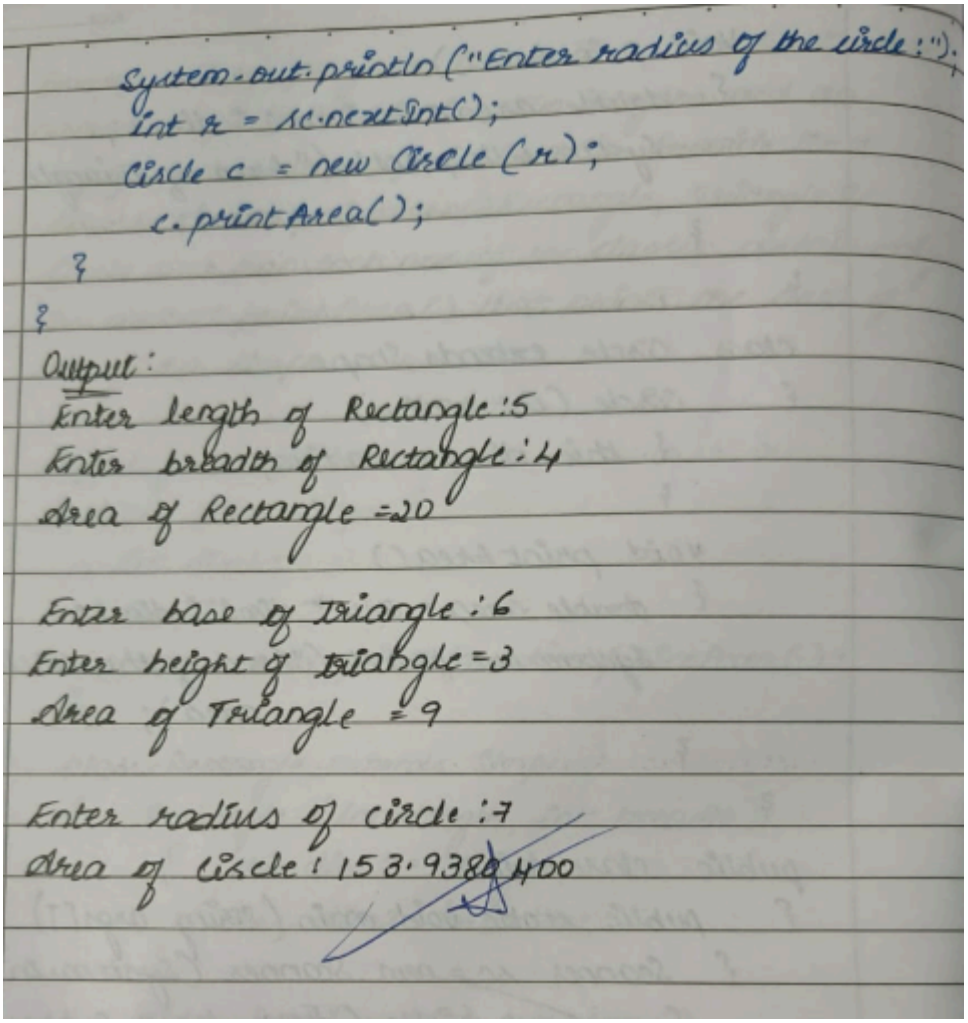
Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea()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

Algorithm:



```
import java.util.Scanner;
abstract class Shape {
    int dim1;
    int dim2;
    abstract void printf(creat); printArea();
}
class Rectangle extends Shape
{   Rectangle (int length, int breadth)
    {   this.dim1 = length;
        this.dim2 = breadth;
    }
    void printf(creat()) printArea()
    {  int area = dim1 * dim2;
       System.out.println("Area of the Rectangle :"
                                    + 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 = 3.14 * dim1 * dim1;
      System.out.println("Area of the circle: " +
                         area);
    }
}
public class Main
{   public static void main(String args[])
    { Scanner sc = new Scanner(System.in);
      System.out.println("Enter length & breadth of the
                          Rectangle: ");
      int l = sc.nextInt();
      int b = sc.nextInt();

      Rectangle rect = new Rectangle(l, b);
      rect.printArea();
      System.out.println("Enter base & height of the
                          Triangle: ");
      int ba = sc.nextInt();
      int h = sc.nextInt();
      Triangle tr = new Triangle(ba, h);
      tr.printArea();
```



Code:

```
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();

    }
}
```

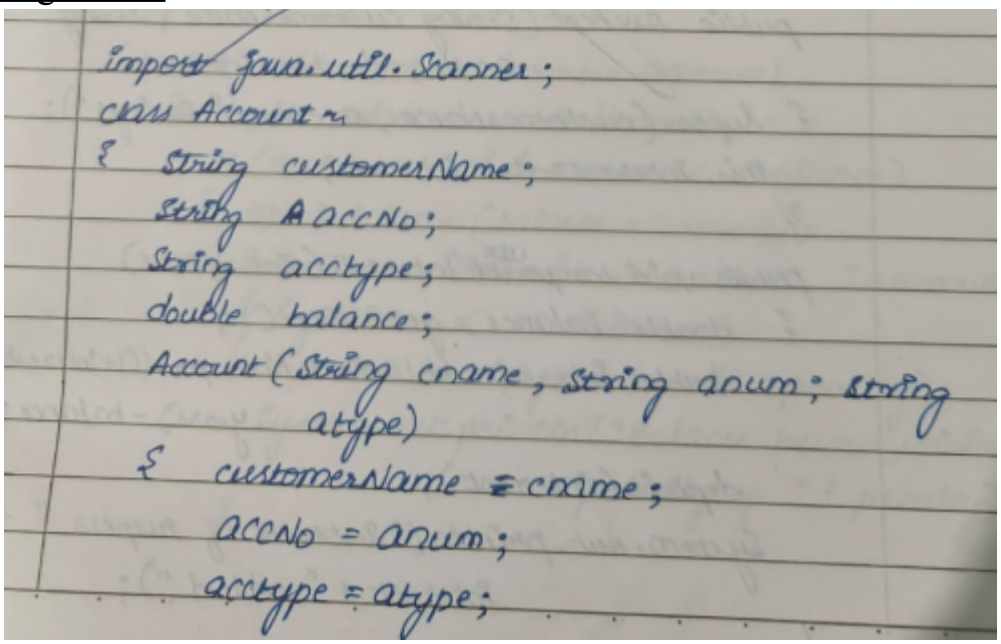
Output:

```
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
```

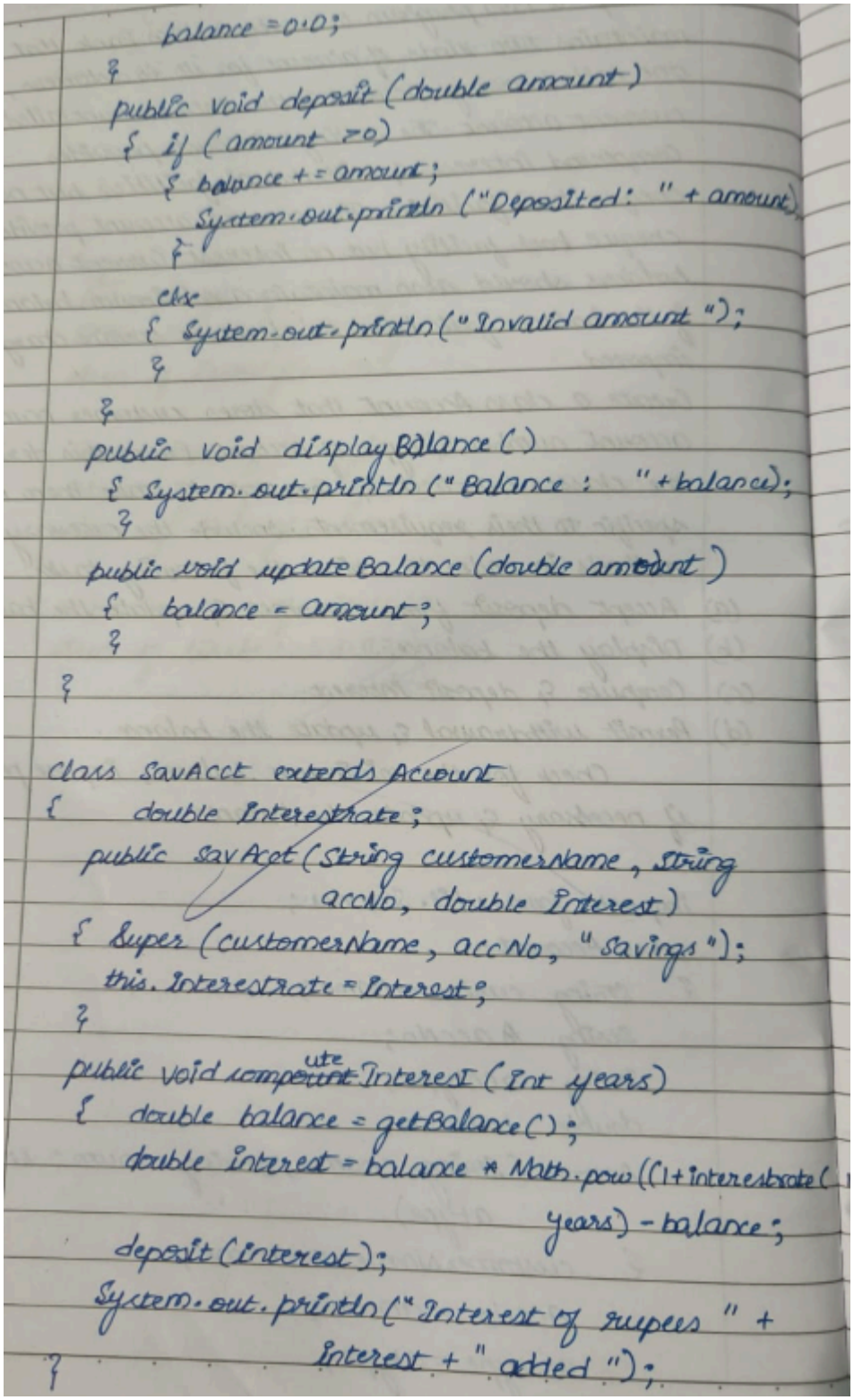
LABORATORY PROGRAM - 5

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.

Algorithm:



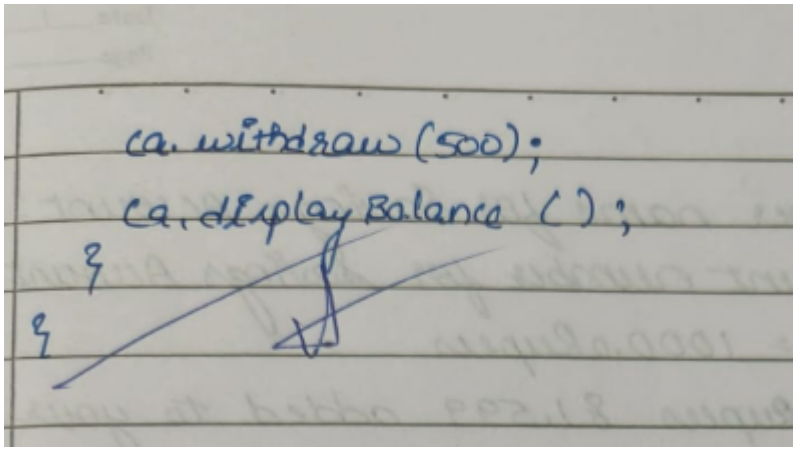
```
import java.util.Scanner;
class Account
{ String customerName;
  String AccNo;
  String acctype;
  double balance;
  Account(String cname, String anum; String
          atype)
  { customerName = cname;
    accNo = anum;
    acctype = atype;
```

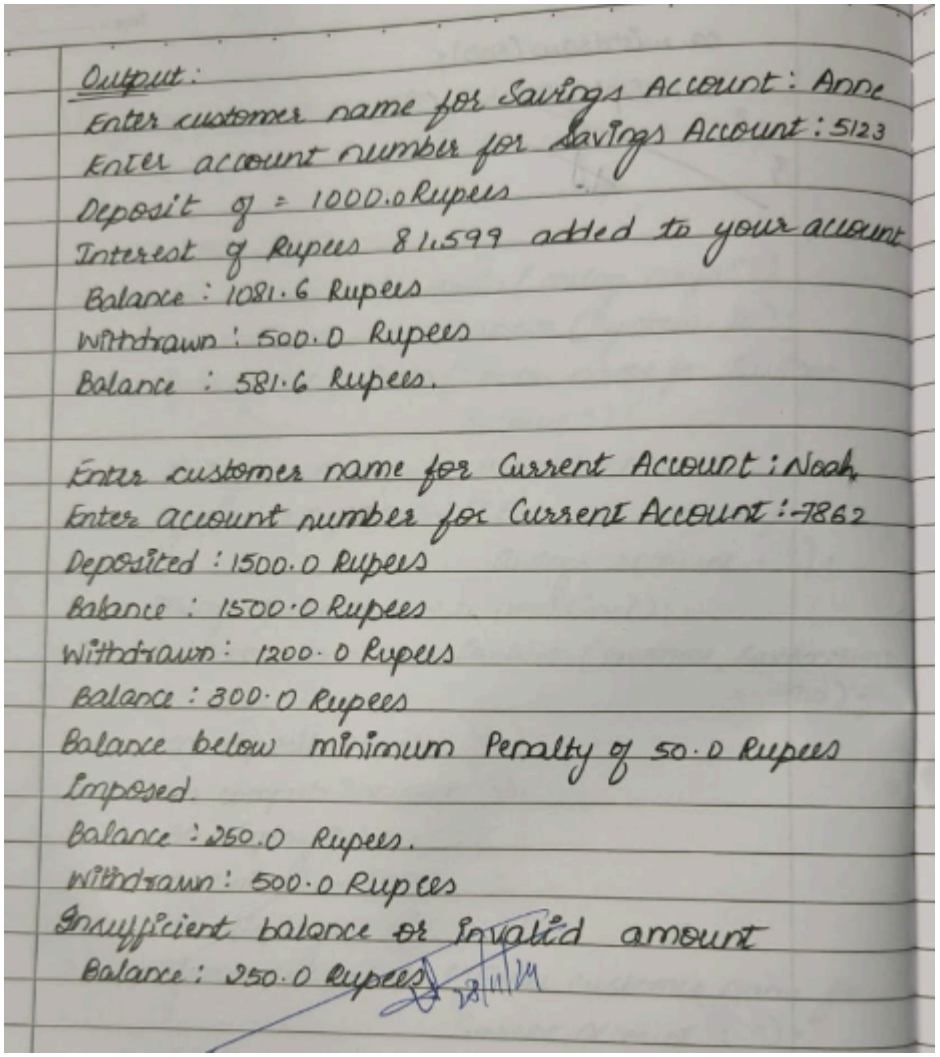


```
        balance = 0.0;
    }
    public void deposit (double amount)
    { if (amount >=0)
      { balance += amount;
        System.out.println ("Deposited: " + amount);
      }
      else
      { System.out.println ("Invalid amount ");
      }
    }
    public void displayBalance ()
    { System.out.println ("Balance :  " + balance);
    }
    public void updateBalance (double amount)
    {  balance = amount;
    }
}

class SavAcct extends Account
{   double Interestrate;
    public SavAcct (String customerName, String
                    accNo, double Interest)
    { Super (customerName, accNo, "Savings");
      this.Interestrate = Interest;
    }
    public void computeInterest (int years)
    { double balance = getBalance();
      double interest = balance * Math.pow((1+interestrate(
                                    years) - balance;
      deposit (interest);
      System.out.println ("Interest of rupees " +
                    Interest + " 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
{  double minBalance;
   double penalty;

   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 (getBalance() - penalty);
         System.out.println ("Balance below minimum
                              penalty of " + penalty);
       }
```

```
else
{ System.out.println("Insufficient balance %/%, or
                      invalid amount ");
}
public class Main
{ public static void main(String args[])
  { Scanner sc =new Scanner(System.in);
    System.out.println("Enter name for Savings
                        account");
    String savName = sc.nextLine();
    System.out.println("Enter the amount for
                        Savings account = ");
    String savAccNum = sc.nextLine();
    SavAcct sa = new SavAcct(SavName, SavAccNum,
                             9.0);
    sa.deposit(1000);
    sa.computeInterest(2);
    sa.displayBalance();
    sa.withdraw(500);
    sa.displayBalance();
    System.out.println("Enter customer name for
                        Current account : ");
    String curName = sc.NextLine();
    System.out.println("Enter account number for
                        current account: ");
    String currAccNum = sc.nextLine();

CurrAcct ca = new CurrAcct(curName, currAccNum,
                           500,50);
ca.deposit(1500);
ca.displayBalance();
Ca.withdraw(1200);
Ca.displayBalance();
```





Code

```
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();
    }
}
```

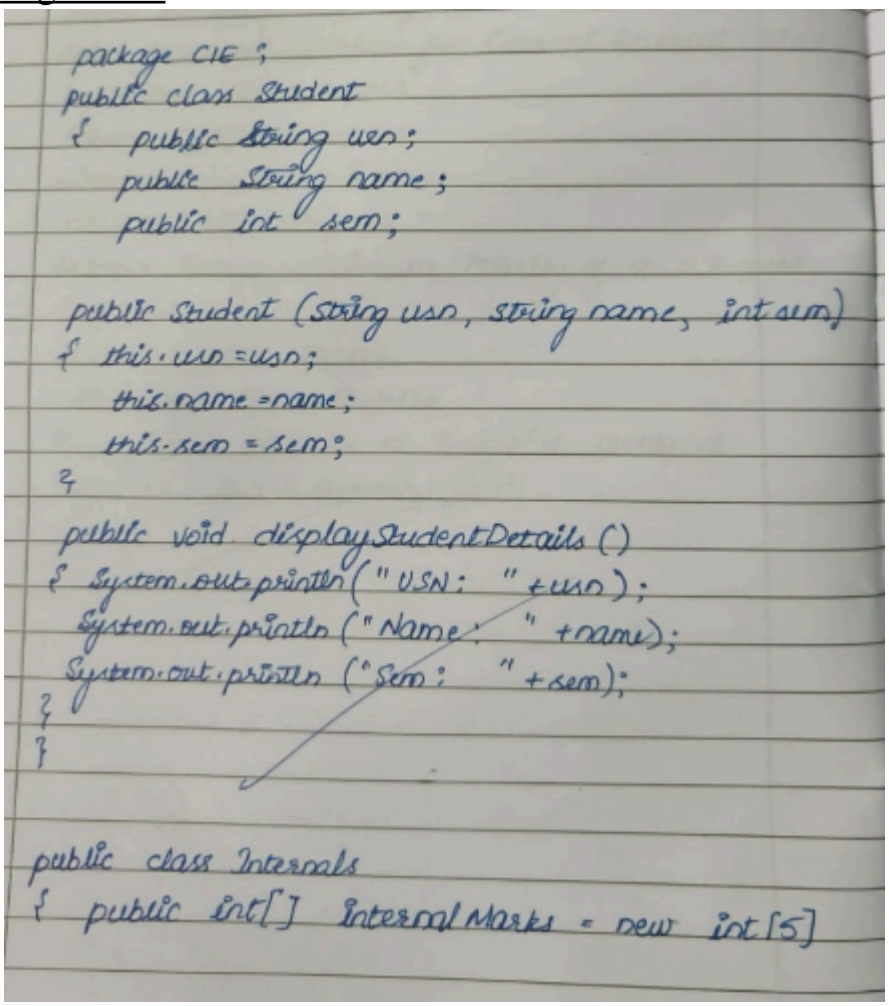
Output:

```
Enter customer name for Savings Account: Saahya K S
Enter account number for Savings Account: 5123
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: Saahya K S
Enter account number for Current Account: 7862
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
```

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.

Algorithm:

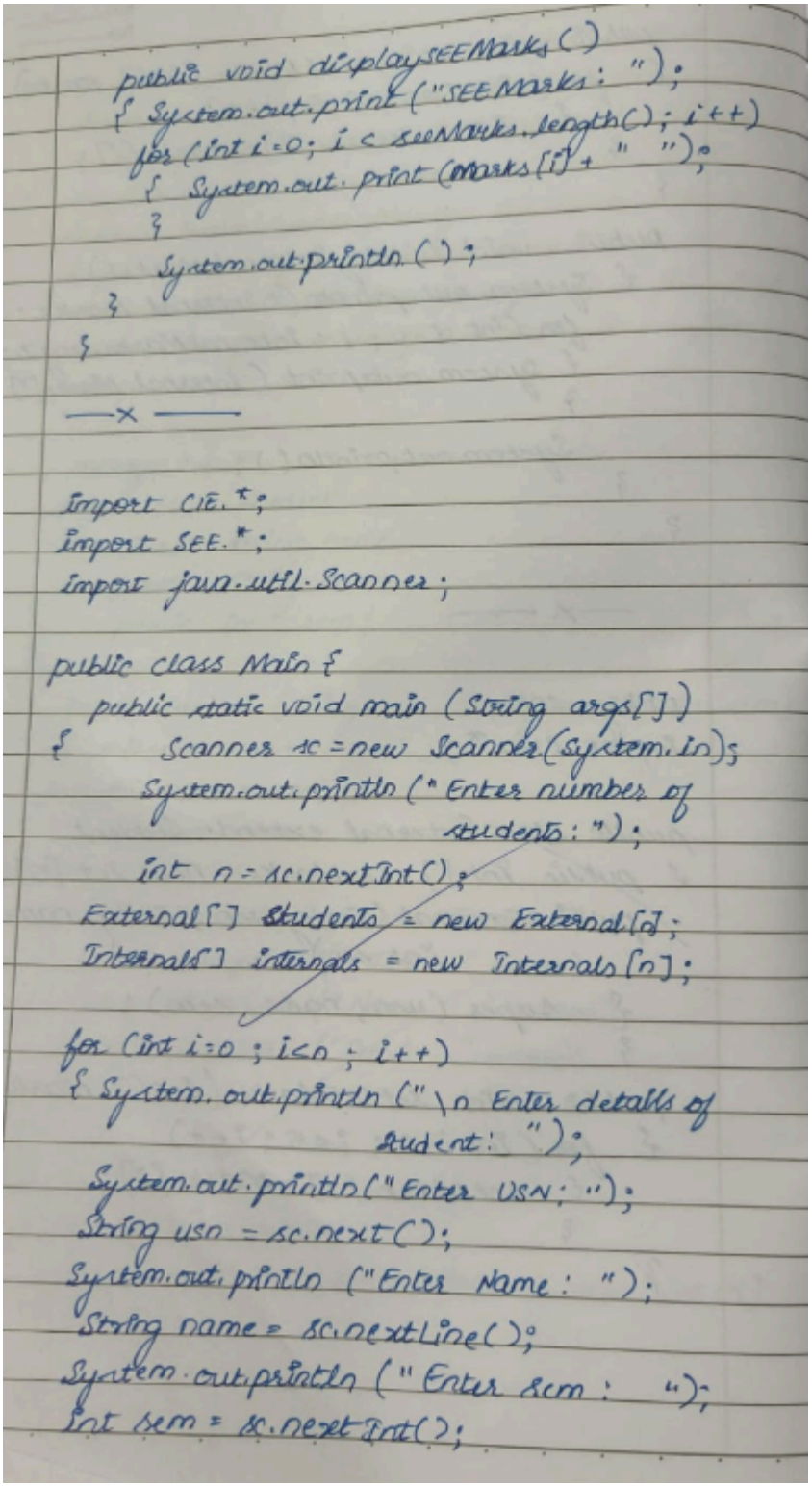


```
package CIE ;
public class Student
{  public String usn;
   public String name;
   public int sem;

public Student (String usn, String name, int sem)
{ this.usn =usn;
   this.name =name;
   this.sem = sem;
}
public void displayStudentDetails ()
{ System.out.println("USN:  "+usn);
 System.out.println("Name:  "+name);
System.out.println("Sem:  "+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.println("Internal Marks : ");
     for (int i=0; i < internalMarks.length; i++)
     {  System.out.print (internalMarks[i] + " ");
     }
     System.out.println();
  }
}
```

—x—

```
package SEE;
import CIE.*;

public class External extends Student
{ 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++)
     {  setMarks[i] = marks[i];
     }
}
```

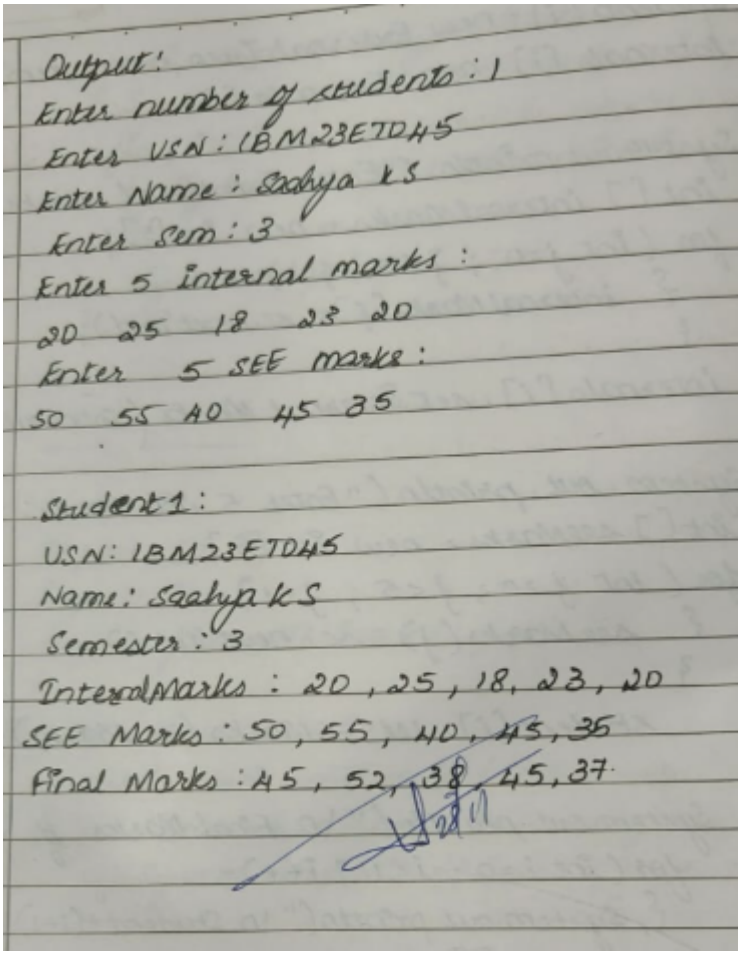


```
    public void displaySEEMarks()
    { System.out.print("SEE Marks: ");
      for(int i=0; i < seeMarks.length(); i++)
      { System.out.print(marks[i] + " ");
      }
      System.out.println();
    }
}
```

—×—

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

public class Main {
    public static void main(String args[])
{       Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of
                                students:");
        int n = sc.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.println("Enter USN: ");
  String usn = sc.next();
  System.out.println("Enter Name: ");
  String name = sc.nextLine();
  System.out.println("Enter Sem: ");
  int sem = sc.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] = sc.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) = sc.nextInt();
  }
   student[i].setSEEMarks(seeMarks);

System.out.println("\n Final Marks of Student");
  for(int i=0; i<n; i++)
  { System.out.println("\n Student"+(i+1)+":");
    students[i].displayStudentDetails();
    Internals[i].displayInternalMarks();
    student[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();
  }
}
```



Code:

```
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 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 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];
        }
    }
```

```
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();

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

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

            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();
            }
            students[i].setSEEMarks(seeMarks);
        }

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

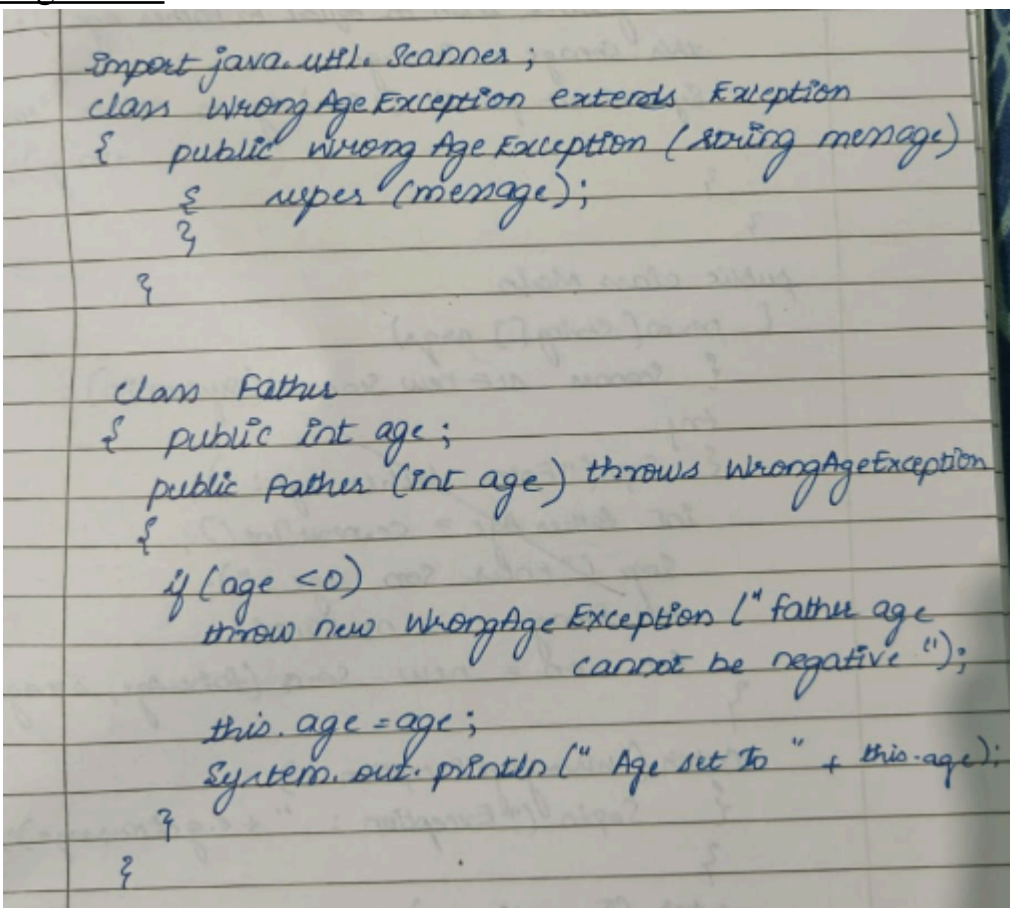
Output:

```
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
```

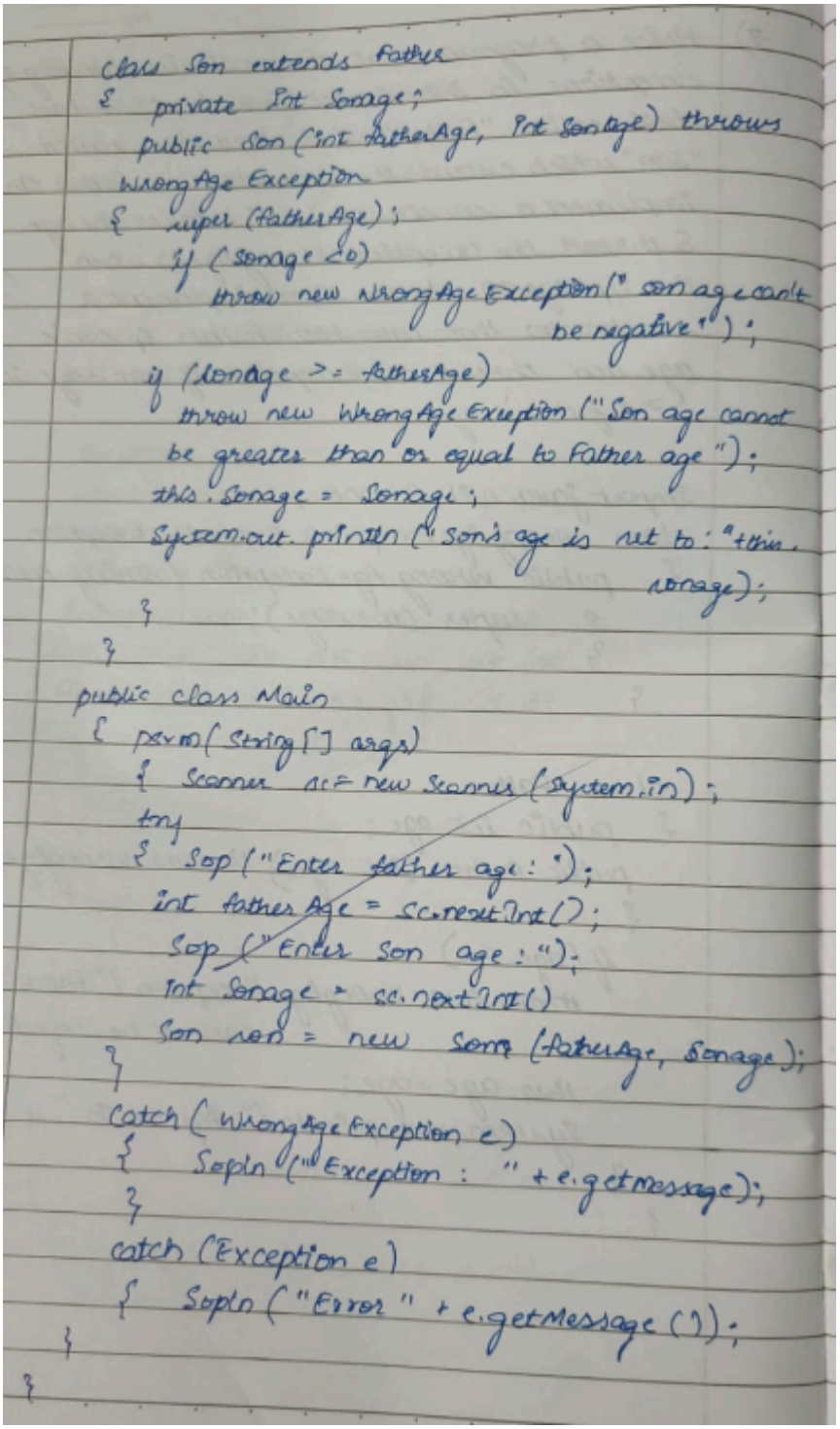
LABORATORY PROGRAM - 7

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.

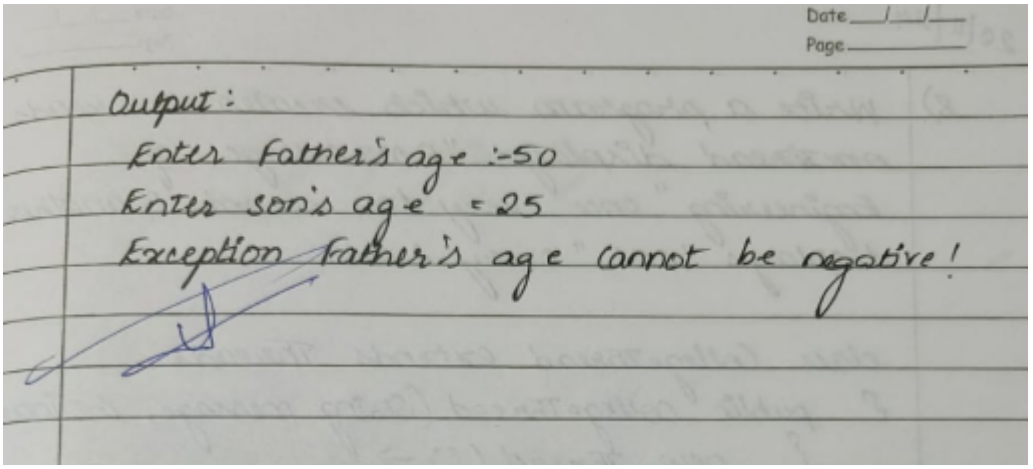
Algorithm:



```
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
  WrongAge Exception
  { super(fatherAge);
    if (Sonage < 0)
      throw new WrongAge Exception("son age can't
                                    be negative");
    if (Sonage >= fatherAge)
      throw new WrongAge Exception("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
{ psvm(String[] args)
  { Scanner sc= new Scanner(System.in);
  try
  { Sop("Enter father age: ");
    int fatherAge = sc.nextInt();
    Sop("Enter Son age: ");
    int Sonage = sc.nextInt()
    Son son = new Son(fatherAge, Sonage);
  }
  catch(WrongAge Exception e)
  { Sopln("Exception : " + e.getMessage);
  }
  catch(Exception e)
  { Sopln("Error " + e.getMessage());
}
}
```



Code:

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

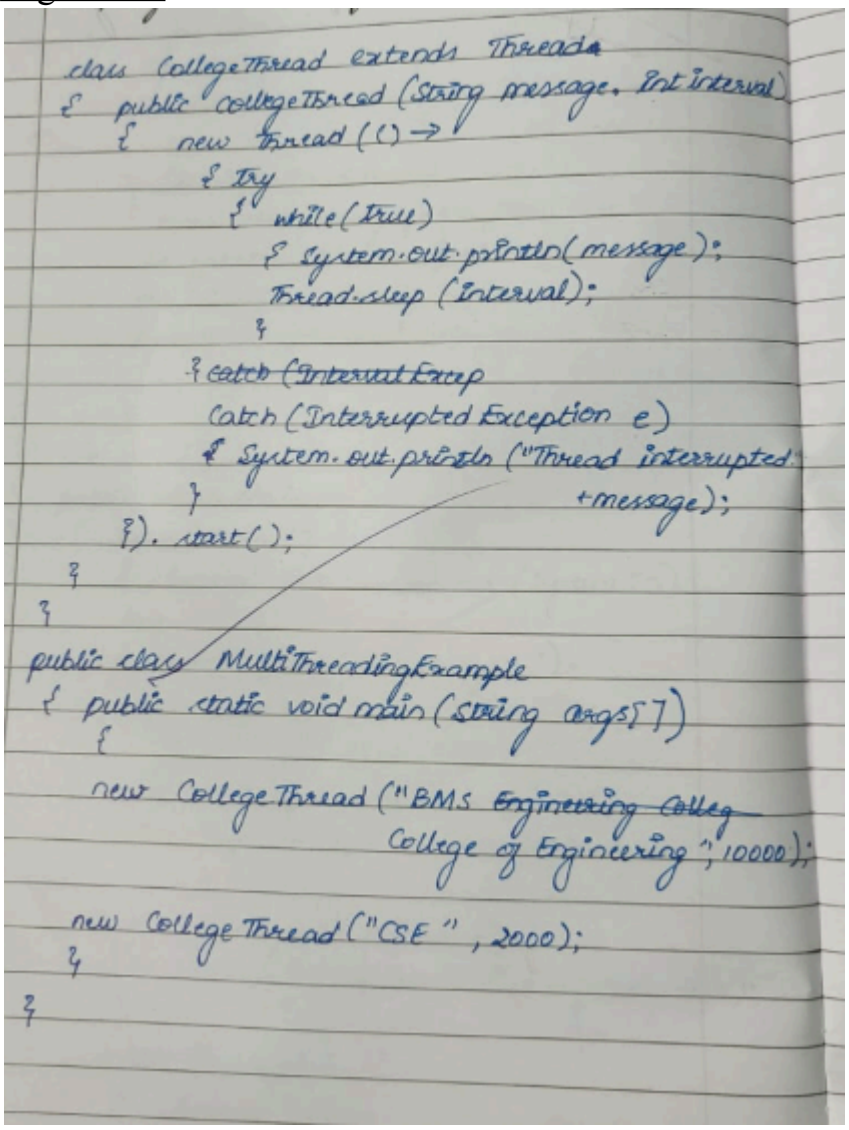
Output:

```
Enter Father's age: -50
Enter Son's age: 25
Exception occurred: Father's age cannot be negative!
Program execution completed.
```

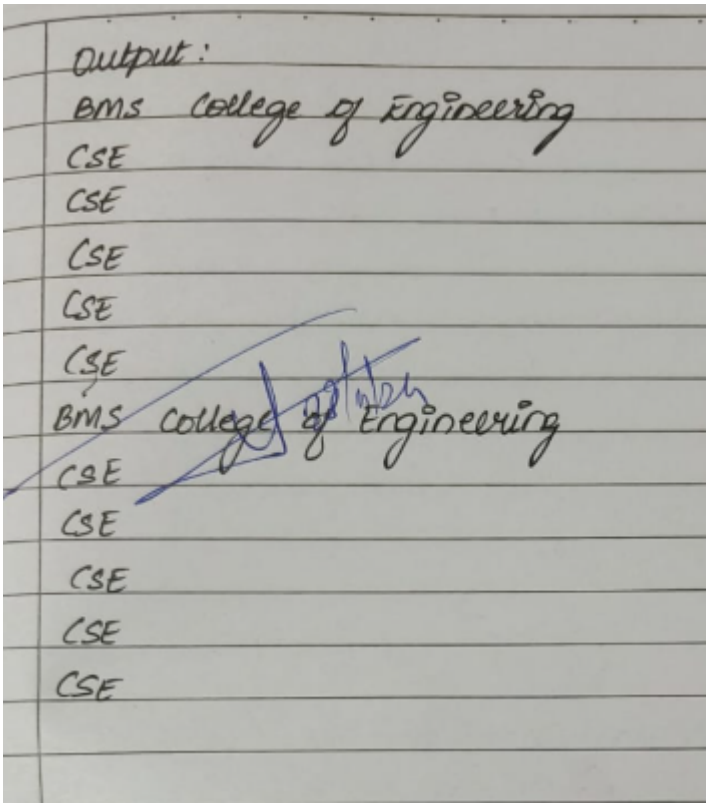
LABORATORY PROGRAM - 8

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.

Algorithm:



```
class CollegeThread extends Thread
{ public CollegeThread (String message, int interval)
  { new Thread (()->
    { try
      { while(true)
        { System.out.println(message);
          Thread.sleep (interval);
        }
      } catch (InterruptedException e)
        { System.out.println ("Thread interrupted:
                              +message);
        }
    }).start();
  }
}
public class MultiThreadingExample
{ public static void main (String args[])
  {
    new CollegeThread ("BMS Engineering Colleg
                        College of Engineering", 10000);
    new CollegeThread ("CSE", 2000);
  }
}
```



Code:

```
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:

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

LABORATORY PROGRAM - 9

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

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

public class DivisionMain1 extends Frame implements ActionListener
{
        TextField
        num1,num2; Button
        dResult;
        Label
        outResult;
        String out="";
        double
        resultNum; int
        flag=0;

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

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

                add(number1)
                ; add(num1);
                add(number2)
```

```
        ; add(num2);
        add(dResult);
        add(outResult
        );

        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(resultN
                        um); repaint();

                }
        }
        catch(NumberFormatException e1)
        {
                flag=1;
```

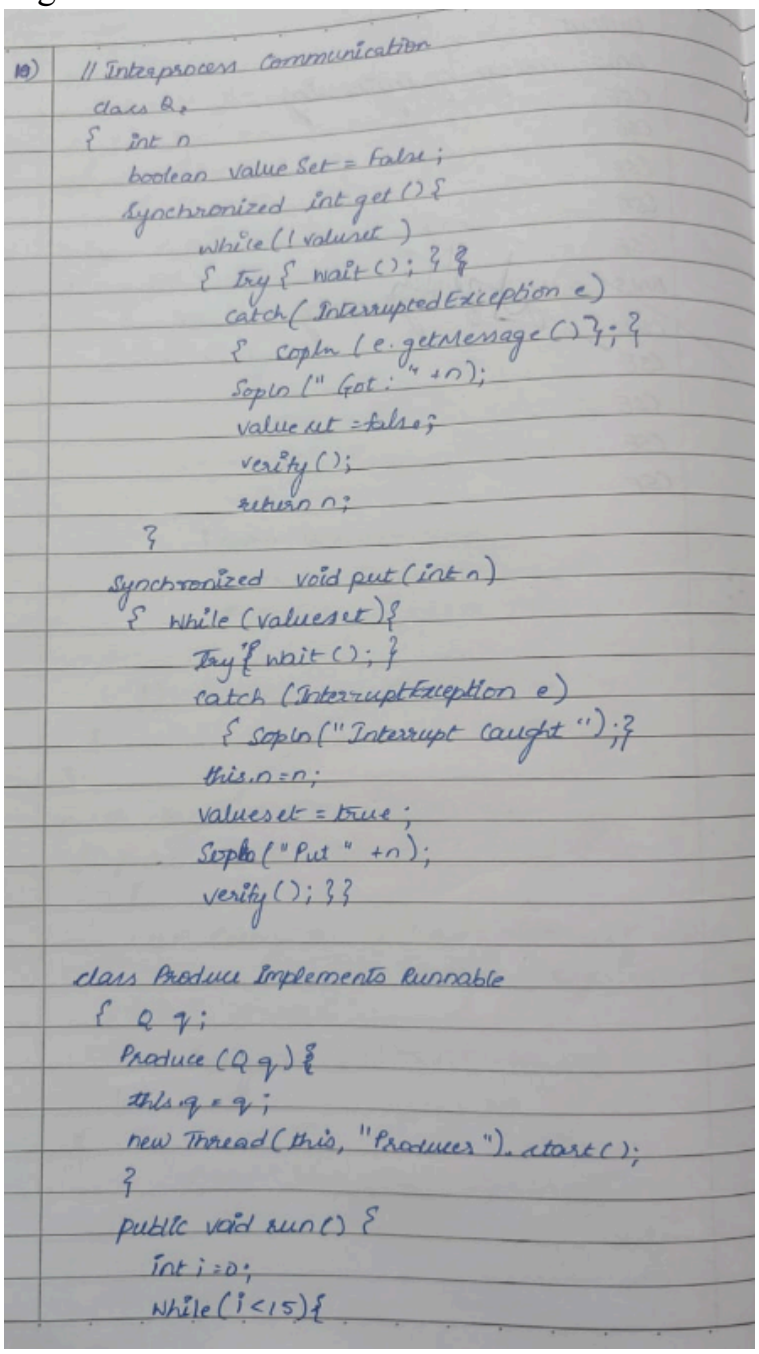
```
                out="Number Format Exception!
                "+e1; repaint();
            }
            catch(ArithmeticException e2)
            {
                flag=1;
                out="Divide by 0 Exception!
                "+e2; repaint();
            }

        }
        public void paint(Graphics g)
        {
            if(flag==0)
            g.drawString(out,outResult.getX()+outResult.getWidth(),outResult
            .getY()+outResult. getHeight()-8);
            else
            g.drawString(out,100
            ,200); flag=0;
        }

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

OUTPUT

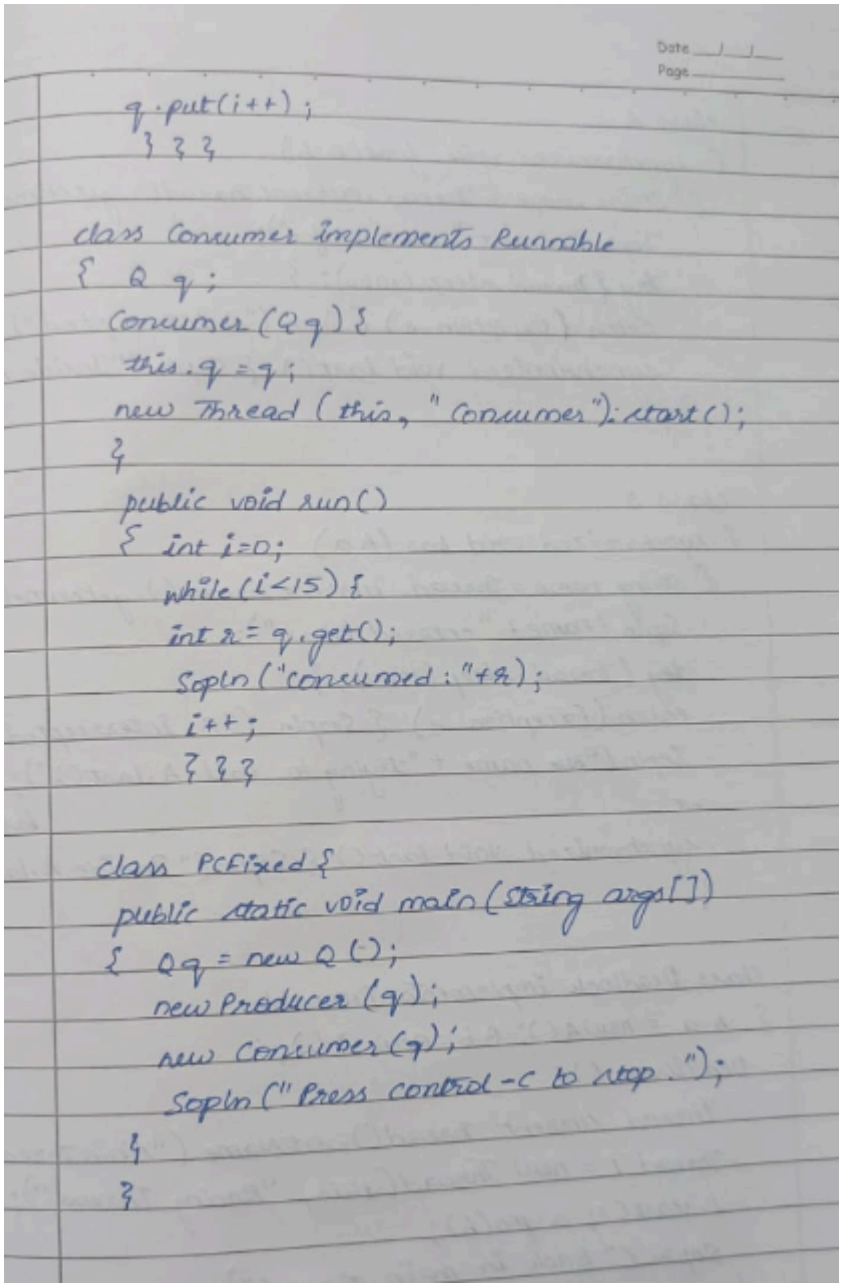
LABORATORY PROGRAM - 10

Demonstrate Interprocess communication and deadlock

Algorithm:



```
10) // Interprocess Communication
class Q.
{ int n
  boolean valueSet = false;
  Synchronized int get(){
      while(!valueset)
      { try{ wait(); }}
       catch( InterruptedException e)
       { Sopln(e.getMessage());}
       Sopln(" Got: " +n);
       valueset = false;
       verify();
       return n;
   }
 Synchronized void put(int n)
  { while(valueset){
      try{ wait(); }
      catch (InterruptException e)
        { Sopln("Interrupt caught");}
      this.n=n;
      valueset = true;
      Sopln("Put " +n);
      verify(); }}

class Produc implements Runnable
 { Q q;
   Produce(Q q){
   this.q = q;
   new Thread(this, "Produces").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();
Sopln("consumed:"+r);
i++;
}}}

class PCFixed{
public static void main(String args[])
{ Q q = new Q();
new Producer(q);
new Consumer(q);
Sopln("Press control-C to stop.");
}
}
```

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

    synchronized int get() {
        while (!valueSet) {
            try {
                System.out.println("\nConsumer waiting\n");
                wait();
            } catch (InterruptedException e) {
                System.out.println("InterruptedException caught");
            }
        }
```

```
        System.out.println("Got: " + n);
        valueSet = false;
        System.out.println("\nIntimate Producer\n");
        notify();
        return n;
    }

    synchronized void put(int n) {
        while (valueSet) {
            try {
                System.out.println("\nProducer waiting\n");
                wait();
            } catch (InterruptedException e) {
                System.out.println("InterruptedException caught");
            }
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        System.out.println("\nIntimate Consumer\n");
        notify();
    }
}

class Producer implements Runnable {
    Q q;

    Producer(Q q) {
        this.q = q;
        new Thread(this, "Producer").start();
    }

    public void run() {
        int i = 0;
        while (i < 15) {
            q.put(i++);
            try {
                Thread.sleep(100);
            } catch (InterruptedException e) {
                System.out.println("Producer interrupted");
            }
        }
    }
}
```

```
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

```
Put: 0

Intimate Consumer


Producer waiting

Got: 0

Intimate Producer

Put: 1

Intimate Consumer

Consumed: 0
Got: 1

Intimate Producer

Consumed: 1

Consumer waiting

Put: 2

Intimate Consumer

Got: 2

Intimate Producer

Consumed: 2

Consumer waiting

Put: 3

Intimate Consumer

Got: 3
```

```
Intimate Producer

Consumed: 3

Consumer waiting

Put: 4

Intimate Consumer

Got: 4

Intimate Producer

Consumed: 4

Consumer waiting

Put: 5

Intimate Consumer

Got: 5

Intimate Producer

Consumed: 5

Consumer waiting

Put: 6

Intimate Consumer

Got: 6

Intimate Producer

Consumed: 6

Consumer waiting

Put: 7

Intimate Consumer

Got: 7

Intimate Producer

Consumed: 7

Consumer waiting

Put: 8

Intimate Consumer

Got: 8

Intimate Producer

Consumed: 8

Consumer waiting

Put: 9
```

```
Intimate Producer

Consumed: 9

Consumer waiting

Put: 10

Intimate Consumer

Got: 10

Intimate Producer

Consumed: 10

Consumer waiting

Put: 11

Intimate Consumer

Got: 11

Intimate Producer

Consumed: 11

Consumer waiting

Put: 12

Intimate Consumer

Got: 12

Intimate Producer

Consumed: 12

Consumer waiting

Put: 13

Intimate Consumer

Got: 13

Intimate Producer

Consumed: 13

Consumer waiting

Put: 14

Intimate Consumer

Got: 14

Intimate Producer

Consumed: 14
```