VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Lavanya B (1BM23CS167)

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 to 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 (23CS3PCOOJ)" carried out by **StudentName (1BM23CS000),** 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. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

Srushti Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Java lab.

INDEX

Name Lavanya.B Std Sec C

Roll No. Subject School/College

School/College Tel. No. Parents Tel. No.

Sl. No.	Date	Title	Page No.	Teacher Sign / Remarks
01	30/9/24	Finding solution for quadratic equation.	10	
02	7/10/24	Calculation SGPA of a student.	10	
03	14/10/24	Printing the details of 'n' books	10	
04	21/10/24	calculation of area of rectangle, triangle, circle.	10	
05	29/10/24	create Class Bank having 2 kinds of accounts.	10	
06	12/11/24	Creation of packages CIE & SEE	10	
07	29/12/24	Exception	10	
08	29/12/24	Threads	10	
09	29/12/24	SwingDemo	10	
10	29/12/24	Interprocess communication & Deadlock.	10	

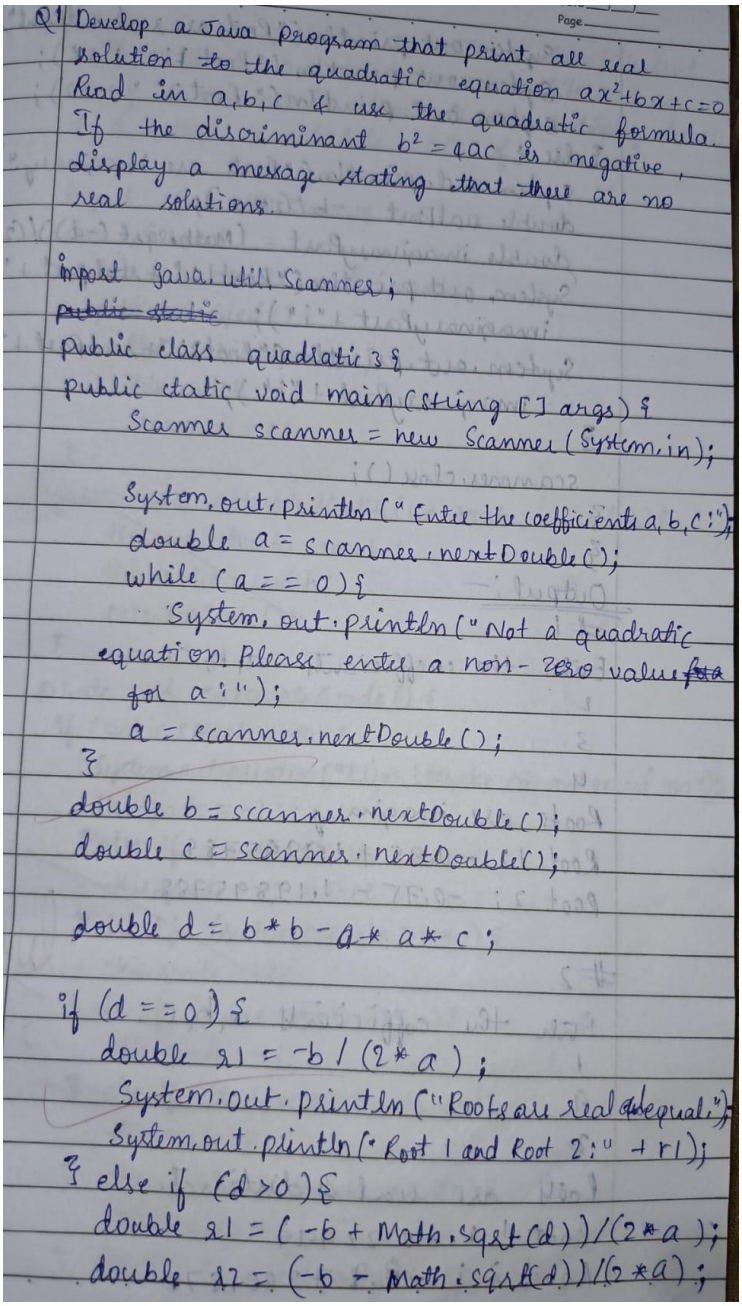
completed

Github Link:
(https://github.com/Lavanya167B/javalab)

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:



Q1 Develop a Java program that print all real solution to the quadratic equation $ax^2+bx+c=0$. Read in a,b,c & use the quadratic formula. If the discriminant b^2-4ac is negative, display a message stating that there are no real solutions.

```
import java.util.Scanner;
public class quadratic3 {
public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    System.out.println("Enter the coefficients a,b,c:");
    double a = scanner.nextDouble();
    while (a==0){
      System.out.println("Not a quadratic
equation. Please enter a non-zero value
for a:");
    a = scanner.nextDouble();
}
double b = scanner.nextDouble();
double c = scanner.nextDouble();

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

if (d==0){
  double r1 = -b/(2*a);
  System.out.println("Roots are real & equal:");
  System.out.println("Root 1 and Root 2:" +r1);
} else if (d>0){
  double r1 = (-b + Math.sqrt(d))/(2*a);
  double r2 = (-b - Math.sqrt(d))/(2*a);
  System.out.println("Roots are real & distinct");
  System.out.println("Root 1: " + r1);
  System.out.println("Root 2: " + r2);
} else {
  System.out.println("Roots are imaginary");
  double realPart = -b/(2*a);
  double imaginaryPart = (Math.sqrt(-d))/(2*a);
  System.out.println("Root 1:" + realPart + "+"
    imaginaryPart + "i");
  System.out.println("Root 2:" + realPart + "-"
    imaginaryPart + "i");
}
  scanner.close();
}
}
```

Output:-

#1

```
Enter the coefficients a,b,c:
2
3
4
Roots are imaginary
Root 1: -0.75 + 1.1989578i
Root 2: -0.75 - 1.1989578i
```

#2

```
Enter the coefficients a,b,c:
1
9
2
Roots are real & distinct.
Root 1: -0.22799
Root 2: -8.772001872
```

Code:

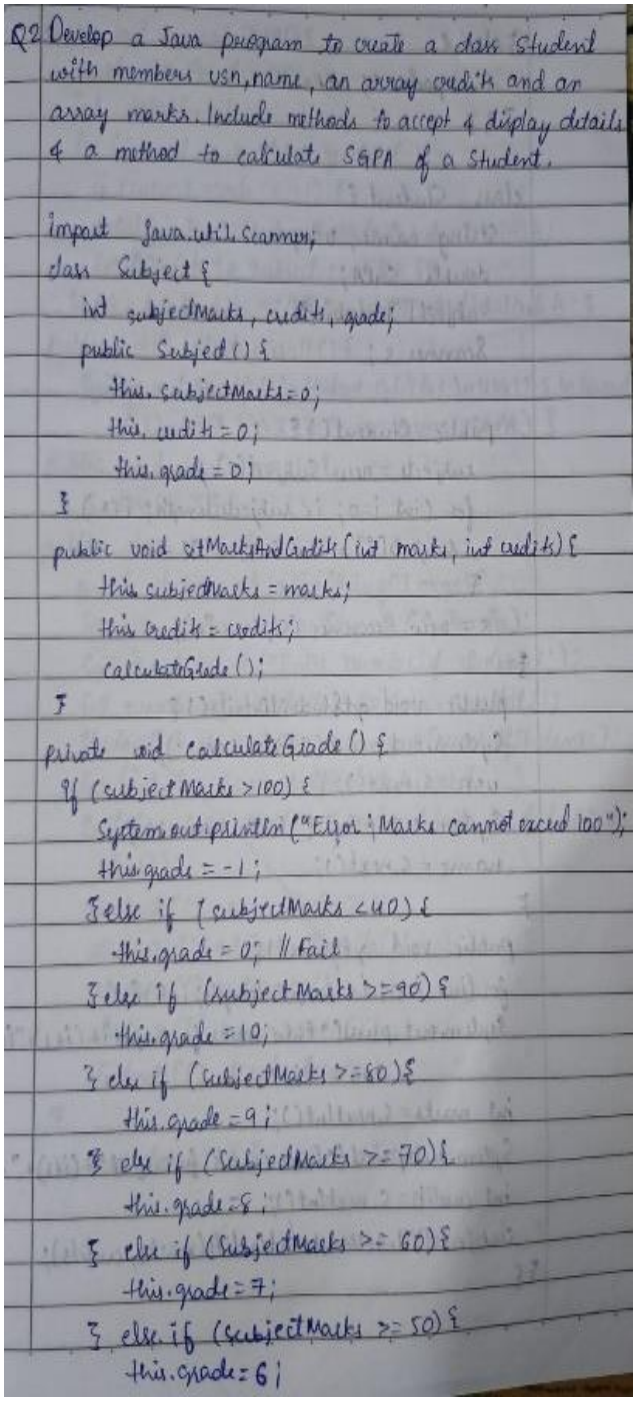
```
import java.util.Scanner;

public class quadratic3{
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      System.out.println("Enter the coefficients a, b, c:");

      double a = scanner.nextDouble();

      while (a == 0) {
         System.out.println("Not a quadratic equation. Please enter a non-zero value for a:");
         a = scanner.nextDouble();
      }

      double b = scanner.nextDouble();
      double c = scanner.nextDouble();

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

      if (d == 0) {
         double r1 = -b / (2 * a);
         System.out.println("Roots are real and equal.");
         System.out.println("Root 1 and Root 2: " + r1);
      } else if (d > 0) {
         double r1 = (-b + Math.sqrt(d)) / (2 * a);
         double r2 = (-b - Math.sqrt(d)) / (2 * a);
         System.out.println("Roots are real and distinct.");
         System.out.println("Root 1: " + r1);
         System.out.println("Root 2: " + r2);
      } else {
         System.out.println("Roots are imaginary.");
         double realPart = -b / (2 * a);
         double imaginaryPart =(Math.sqrt(-d))/ (2 * a);
         System.out.println("Root 1: " + realPart + " + " + imaginaryPart + "i");
         System.out.println("Root 2: " + realPart + " - " + imaginaryPart + "i");
      }

      scanner.close();
   }
}
```

```
D:\1BM23CS167>javac quadratic3.java

D:\1BM23CS167>java quadratic3
Enter the coefficients a, b, c:
2
3
4
Roots are imaginary.
Root 1: -0.75 + 1.19895788082817981
Root 2: -0.75 - 1.19895788082817981

D:\1BM23CS167>javac quadratic3.java

D:\1BM23CS167>java quadratic3
Enter the coefficients a, b, c:
1
9
2
Roots are real and distinct.
Root 1: -0.22799812734123348
Root 2: -8.772001872658766

D:\1BM23CS167>javac quadratic3.java

D:\1BM23CS167>java quadratic3
Enter the coefficients a, b, c:
1
2
1
Roots are real and equal.
Root 1 and Root 2: -1.0

D:\1BM23CS167>javac quadratic3.java

D:\1BM23CS167>java quadratic3
Enter the coefficients a, b, c:
0
Not a quadratic equation. Please enter a non-zero value for a:
```

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:



```
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 & display details & a method to calculate SGPA of a student.

import java.util.Scanner;
class Subject {
    int subjectMarks, credits, grade;
    public Subject() {
        this.subjectMarks = 0;
        this.credits = 0;
        this.grade = 0;
    }
    public void setMarksAndCredits(int marks, int credits) {
        this.subjectMarks = marks;
        this.credits = credits;
        calculateGrade();
    }
    private void calculateGrade() {
        if (subjectMarks > 100) {
            System.out.println("Error: Marks cannot exceed 100");
            this.grade = -1;
        } else if (subjectMarks < 40) {
            this.grade = 0; // Fail
        } else if (subjectMarks >= 90) {
            this.grade = 10;
        } else if (subjectMarks >= 80) {
            this.grade = 9;
        } else if (subjectMarks >= 70) {
            this.grade = 8;
        } else if (subjectMarks >= 60) {
            this.grade = 7;
        } else if (subjectMarks >= 50) {
            this.grade = 6;
        } else {
            this.grade = 5; // Pass
        }
    }
}

class Student {
    String name, usn;
    double SGPA;
    Subject[] subjects;
    Scanner s;

    public Student() {
        subjects = new Subject[8];
        for (int i=0; i < subjects.length; i++) {
            subjects[i] = new Subject();
        }
        s = new Scanner(System.in);
    }
    public void getStudentDetails() {
        System.out.print("Enter USN: ");
        usn = s.next();
        System.out.print("Enter Name: ");
        name = s.next();
    }
    public void getMarks() {
        for (int i=0; i < subjects.length; i++) {
            System.out.print("Enter marks for subject " + (i+1) + " (out of 100): ");
            int marks = s.nextInt();
            System.out.print("Enter credits for subject " + (i+1) + ": ");
            int credits = s.nextInt();
            subjects[i].setMarksAndCredits(marks, credits);
        }
    }
}
```

```
public void computeSGPA() {
    double totalPoints = 0.0;
    int totalCredits = 0;
    for (Subject subject : subjects) {
        if (subject.grade != -1) {
            totalPoints += subject.grade * subject.credits;
            totalCredits += subject.credits; } }
    SGPA = totalCredits == 0 ? 0.0 : totalPoints / totalCredits; }
public void displayResult() {
    System.out.printf("\n Student Details: \n USN: %s \n Name:
        %s \n SGPA : %.2f \n", usn, name, SGPA); }
public void closeScanner() {
    s.close(); } }
public class Main {
    public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter number of students:");
    int numberOfStudents = scanner.nextInt();
    Student[] students = new Student[numberOfStudents];
    for (int i=0; i<numberOfStudents; i++) {
    System.out.println("\n Entering details for student" +
        (i+1));
    students[i] = new Student();
    students[i].getStudentDetails();
    students[i].getMarks();
    students[i].computeSGPA();
    students[i].displayResult();
}
scanner.close(); } }
```

Output :-

```
Enter the number of students : 2

Entering details for Student 1
Enter USN : 1bm23cs167
Enter Name : lavanyab
Enter marks for subject 1 (out of 100) : 90
Enter credits for subject 1 : 4
Enter marks for subject 2 (out of 100) : 80
Enter credits for subject 2 : 4
Enter marks for subject 3 (out of 100) : 89
Enter credits for subject 3 : 3
Enter marks for subject 4 (out of 100) : 85
Enter credits for subject 4 : 3
Enter marks for subject 5 (out of 100) : 85
Enter credits for subject 5 : 3
Enter marks for subject 6 (out of 100) : 99
Enter credits for subject 6 : 1
Enter marks for subject 7 (out of 100) : 99
Enter credits for subject 7 : 1
Enter marks for subject 8 (out of 100) : 74
Enter credits for subject 8 : 1

Student Details:
USN : 1bm23cs167
Name : lavanyab
SGPA : 9.25

Entering details for Student 2
Enter USN : 1bm23cs169
Enter Name : Lakshmi
Enter marks for subject 1 (out of 100):
Enter credits for subject 1 : 4
Enter marks for subject 2 (out of 100):
Enter credits for subject 2 : 4
Enter marks for subject 3 (out of 100):
Enter credits for subject 3 : 3
Enter marks for subject 4 (out of 100):
Enter credits for subject 4 : 3
Enter marks for subject 5 (out of 100):
Enter credits for subject 5 : 3
Enter marks for subject 6 (out of 100):
Enter credits for subject 6 : 1
Enter marks for subject 7 (out of 100):
Enter credits for subject 7 : 1
Enter marks for subject 8 (out of 100):
Enter credits for subject 8 : 1
```

14/10/24

Code:

```
import java.util.Scanner;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
class Subject {
    int subjectMarks;
    int credits;
    int grade;
    public Subject() {
        this.subjectMarks = 0;
        this.credits = 0;
        this.grade = 0;
    }
    public void calculateGrade() {
        if (subjectMarks > 100) {
            System.out.println("Error: Marks cannot exceed 100.");
            grade = -1;
        } else if (subjectMarks < 40) {
            grade = 0;
        } else if (subjectMarks >= 90) {
            grade = 10;
        } else if (subjectMarks >= 80) {
            grade = 9;
        } else if (subjectMarks >= 70) {
            grade = 8;
        } else if (subjectMarks >= 60) {
            grade = 7;
        } else if (subjectMarks >= 50) {
            grade = 6;
        } else {
            grade = 5;
        }}}
class Student {
    String name;
    String usn;
    double SGPA;
    Subject[] subjects;
    Scanner s;
    Student() {
        subjects = new Subject[8];
```

```
        for (int i = 0; i < subjects.length; i++) {
            subjects[i] = new Subject();
        }
        s = new Scanner(System.in);
    }
    public void getStudentDetails() {
        System.out.print("Enter USN: ");
        this.usn = s.nextLine();
        System.out.print("Enter Name: ");
        this.name = s.nextLine();
    }
    public void getMarks() {
        for (int i = 0; i < subjects.length; i++) {
            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            subjects[i].subjectMarks = s.nextInt();
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
            subjects[i].credits = s.nextInt();
            subjects[i].calculateGrade();
            if (subjects[i].grade == -1) {
                System.out.println("Please re-enter valid marks and credits.");
                i--;
                continue;
            }}}
    public void computeSGPA() {
        double totalCredits = 0;
        double totalPoints = 0;
        for (Subject subject : subjects) {
            totalCredits += subject.credits;
            totalPoints += subject.grade * subject.credits;
        }
        SGPA = totalCredits == 0 ? 0 : totalPoints / totalCredits;
    }}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
       List<Student> students = new ArrayList<>();
        System.out.print("Enter the number of students: ");
        int n = scanner.nextInt();
        scanner.nextLine(); // Consume newline
        for (int i = 0; i < n; i++) {
            System.out.println("\nEntering details for student " + (i + 1));
```

```
            Student student = new Student();
            student.getStudentDetails();
            student.getMarks();
            student.computeSGPA();
            students.add(student);
        }
        System.out.println("\nStudent Details:");
        for (Student student : students) {
            System.out.printf("Name: %s\nUSN: %s\nSGPA: %.2f\n", student.name, student.usn,
student.SGPA);
        }
        scanner.close();
    }
```

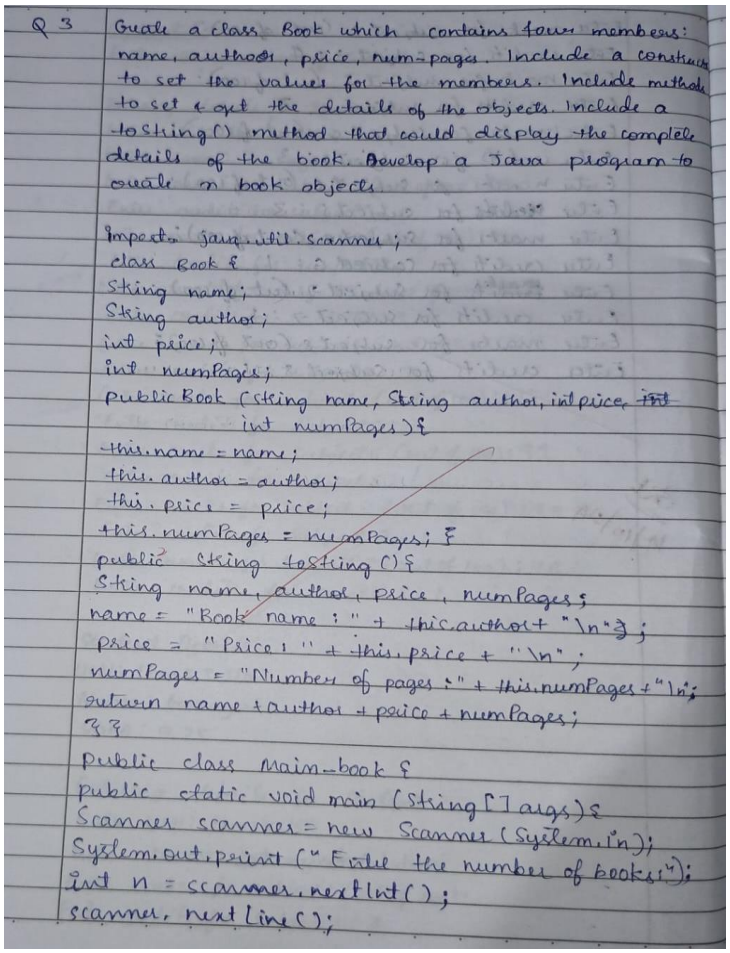
```
D:\1BM23CS167>java sgpa
Enter the number of students: 2

Entering details for student 1
Enter USN: 1BM23CS167
Enter Name: lavanya b
Enter marks for subject 1: 88
Enter credits for subject 1: 3
Enter marks for subject 2: 91
Enter credits for subject 2: 4
Enter marks for subject 3: 91
Enter credits for subject 3: 3
Enter marks for subject 4: 96
Enter credits for subject 4: 3
Enter marks for subject 5: 89
Enter credits for subject 5: 3
Enter marks for subject 6: 88
Enter credits for subject 6: 4
Enter marks for subject 7: 89
Enter credits for subject 7: 1
Enter marks for subject 8: 87
Enter credits for subject 8: 1

Entering details for student 2
Enter USN: 1BM23CS168
Enter Name: lavanya SS
Enter marks for subject 1: 89
Enter credits for subject 1: 4
Enter marks for subject 2: 89
Enter credits for subject 2: 4
Enter marks for subject 3: 98
Enter credits for subject 3: 3
Enter marks for subject 4: 84
Enter credits for subject 4: 3
```

```
Enter marks for subject 5: 86
Enter credits for subject 5: 3
Enter marks for subject 6: 85
Enter credits for subject 6: 1
Enter marks for subject 7: 85
Enter credits for subject 7: 1
Enter marks for subject 8: 84
Enter credits for subject 8: 1

Student Details:
Name: lavanya b
USN: 1BM23CS167
SGPA: 9.45
Name: lavanya SS
USN: 1BM23CS168
SGPA: 9.15

D:\1BM23CS167>
```

Program 3

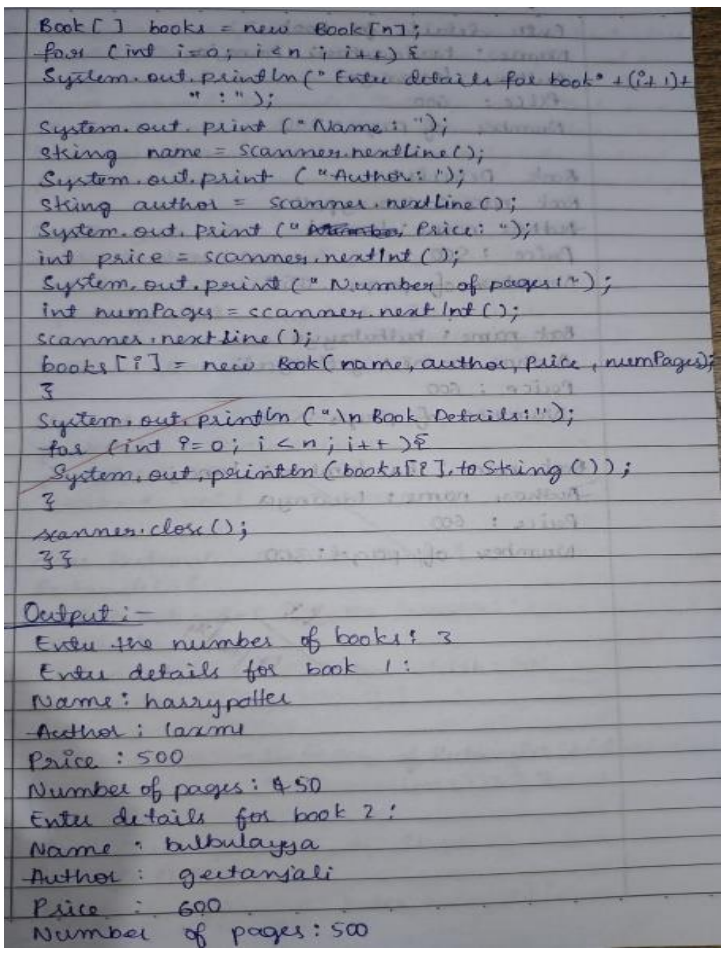
Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString( ) method that could
display the complete details of the book. Develop a Java
program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

Algorithm:



Q 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 & get the details of the objects. Include a toString() method that could display the complete details of the book. Develop a Java program to create n book objects.

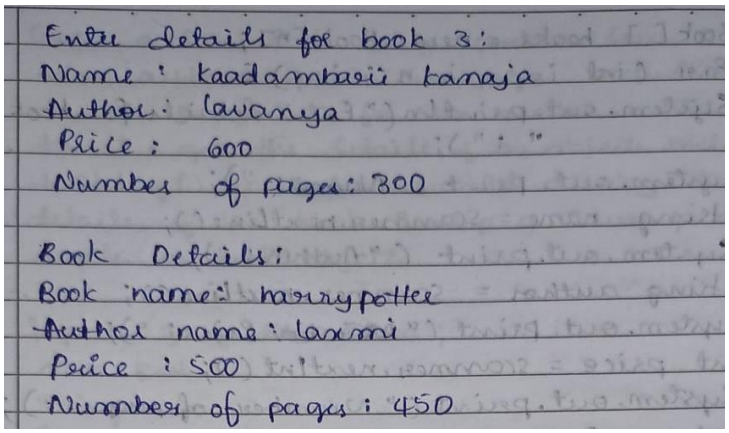
```
import java.util.Scanner;
class Book {
String name;
String author;
int price;
int numPages;
public Book (String name, String author, int price,
        int numPages){
this.name = name;
this.author = author;
this.price = price;
this.numPages = numPages; }
public String toString(){
String name, author,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



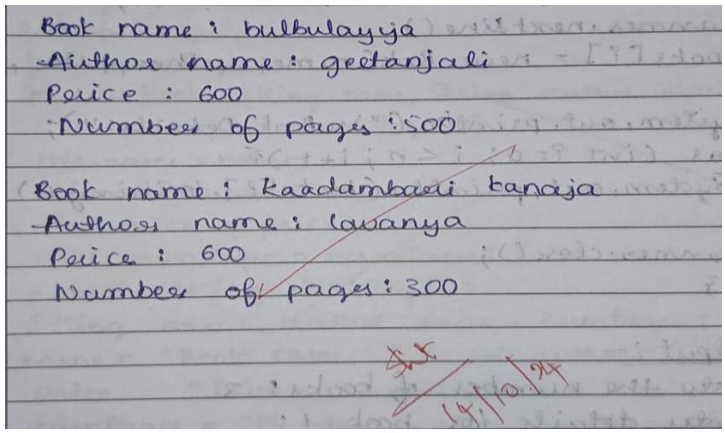
```
Book[] books = new Book[n];
for (int i=0; i<n; i++){
System.out.println("Enter details for book" +(i+1)+
        " :");
System.out.print ("Name : ");
String name = scanner.nextLine();
System.out.print ("Author: ");
String author = scanner.nextLine();
System.out.print ("Price: ");
int price = scanner.nextInt();
System.out.print ("Number of pages : ");
int numPages = scanner.nextInt();
scanner.nextLine();
books[i] = new Book(name, author, price, numPages);
}
System.out.println ("\n Book Details:");
for (int i=0; i<n; i++){
System.out.println(books[i].toString());
}
scanner.close();
}}
```

Output :-

```
Enter the number of books: 3
Enter details for book 1:
Name: harrypotter
Author : laxmi
Price : 500
Number of pages: 450
Enter details for book 2:
Name : bulbulayya
Author : geetanjali
Price : 600
Number of pages: 500
```



```
Enter details for book 3:
Name : kaadambaii kanaja
Author: lavanya
Price : 600
Number of pages: 300

Book Details:
Book name: harrypotter
Author name: laxmi
Price : 500
Number of pages : 450
```



```
Book name : bulbulayya
Author name : geetanjali
Price : 600
Number of pages : 500

Book name : kaadambaii kanaja
Author name : lavanya
Price : 600
Number of pages : 300
```

Code:

```
import java.util.Scanner;
class Book {
String name;
String author;
int price;
int numPages;
public Book(String name, String author, int price, int numPages) {
this.name = name;
this.author = author;
this.price = price;
this.numPages = numPages;
}
public String toString() {
String name, author, price, numPages;
name = "Book name: " + this.name + "\n";
author = "Author name: " + this.author + "\n";
price = "Price: " + this.price + "\n";
numPages = "Number of pages: " + this.numPages + "\n";
return name + author + price + numPages;
}}
public class Main_book {
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("Enter details for book " + (i + 1) + ":");
System.out.print("Name: ");
String name = scanner.nextLine();
System.out.print("Author: ");
String author = scanner.nextLine();
System.out.print("Price: ");
int price = scanner.nextInt();
System.out.print("Number of pages: ");
int numPages = scanner.nextInt();
scanner.nextLine();
books[i] = new Book(name, author, price, numPages);
}
```

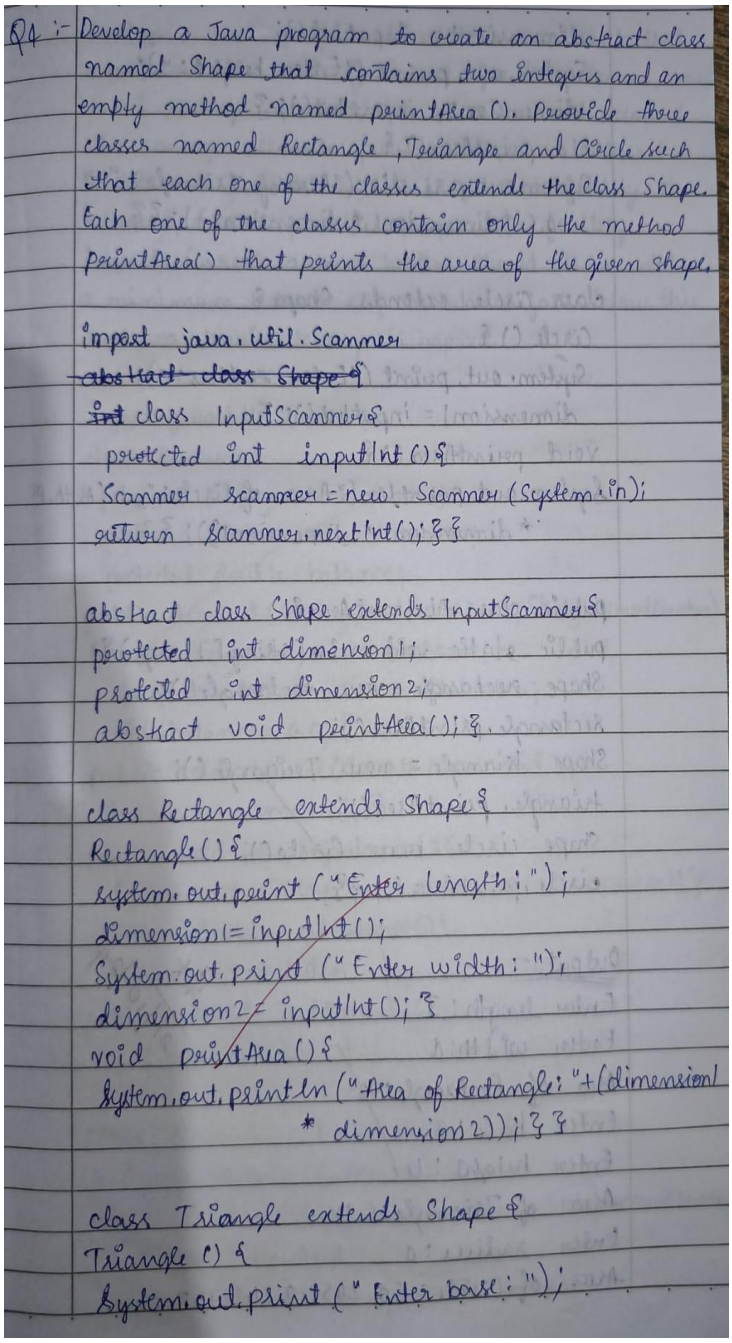
```
System.out.println("\nBook Details:");
for (int i = 0; i < n; i++) {
System.out.println(books[i].toString());
}
scanner.close();
}}
```

```
D:\1BM23CS167>javac Main_book.java

D:\1BM23CS167>java Main_book
Enter the number of books: 3
Enter details for book 1:
Name: harrypotter
Author: laxmi
Price: 500
Number of pages: 450
Enter details for book 2:
Name: bulbulayya
Author: geetanjali
Price: 600
Number of pages: 500
Enter details for book 3:
Name: kaadambari kanaja
Author: lavanya
Price: 600
Number of pages: 300

Book Details:
Book name: harrypotter
Author name: laxmi
Price: 500
Number of pages: 450

Book name: bulbulayya
Author name: geetanjali
Price: 600
Number of pages: 500

Book name: kaadambari kanaja
Author name: lavanya
Price: 600
Number of pages: 300


D:\1BM23CS167>
```

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:



Q4 :- 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.

```
import java.util.Scanner
abstract class Shape{
int class InputScanner{
  protected int inputInt(){
  Scanner scanner = new Scanner(System.in);
  return scanner.nextInt();}}

abstract class Shape extends InputScanner{
protected int dimension1;
protected int dimension2;
abstract void printArea();}

class Rectangle extends Shape{
Rectangle(){
system.out.print("Enter length:");
dimension1=inputInt();
System.out.print("Enter width:");
dimension2= inputInt();}
void printArea(){
System.out.println("Area of Rectangle:"+(dimension1
        * dimension2));}}

class Triangle extends Shape{
Triangle(){
System.out.print("Enter base:");
dimension1 = inputInt();
System.out.print("Enter height:");
dimension2= inputInt();}
void printArea(){
System.out.println("Area of Triangle:" +
  0.5 *dimension1 * dimension2));}}

class Circle extends Shape{
Circle(){
System.out.print("Enter radius:");
dimension1 = inputInt();}
void printArea(){
System.out.println("Area of Circle:"+(Math.PI
    * dimension1 * dimension1));}}

public class AbstractArea{
public static void main(String[] args){
Shape rectangle = new Rectangle();
Rectangle.printArea();
Shape triangle = new Triangle();
triangle.printArea();
Shape circle =new Circle();
circle.printArea();}}
```

Output :-

```
Enter length: 3
Enter width:4
Area of Rectangle:12
Enter base:3
Enter height:4
Area of Triangle:6.0
Enter radius:4
Area of Circle:50.26548245743669
```

21/10/24

Code:

```
import java.util.Scanner;
class InputScanner {
 protected int inputInt() {
 Scanner scanner = new Scanner(System.in);
 return scanner.nextInt();
   }}
abstract class Shape extends InputScanner {
   protected int dimension1;
   protected int dimension2;

   abstract void printArea();
}
class Rectangle extends Shape {
   Rectangle() {
  System.out.print("Enter length: ");
  dimension1 = inputInt();
  System.out.print("Enter width: ");
  dimension2 = inputInt();
   }
   void printArea() {
      System.out.println("Area of Rectangle: " + (dimension1 * dimension2));
   }}
class Triangle extends Shape {
   Triangle() {
      System.out.print("Enter base: ");
      dimension1 = inputInt();
      System.out.print("Enter height: ");
      dimension2 = inputInt();
   }
   void printArea() {
      System.out.println("Area of Triangle: " + (0.5 * dimension1 * dimension2));
   }}
class Circle extends Shape {
   Circle() {
      System.out.print("Enter radius: ");
      dimension1 = inputInt();
   }
   void printArea() {
      System.out.println("Area of Circle: " + (Math.PI * dimension1 * dimension1));
   }}
```

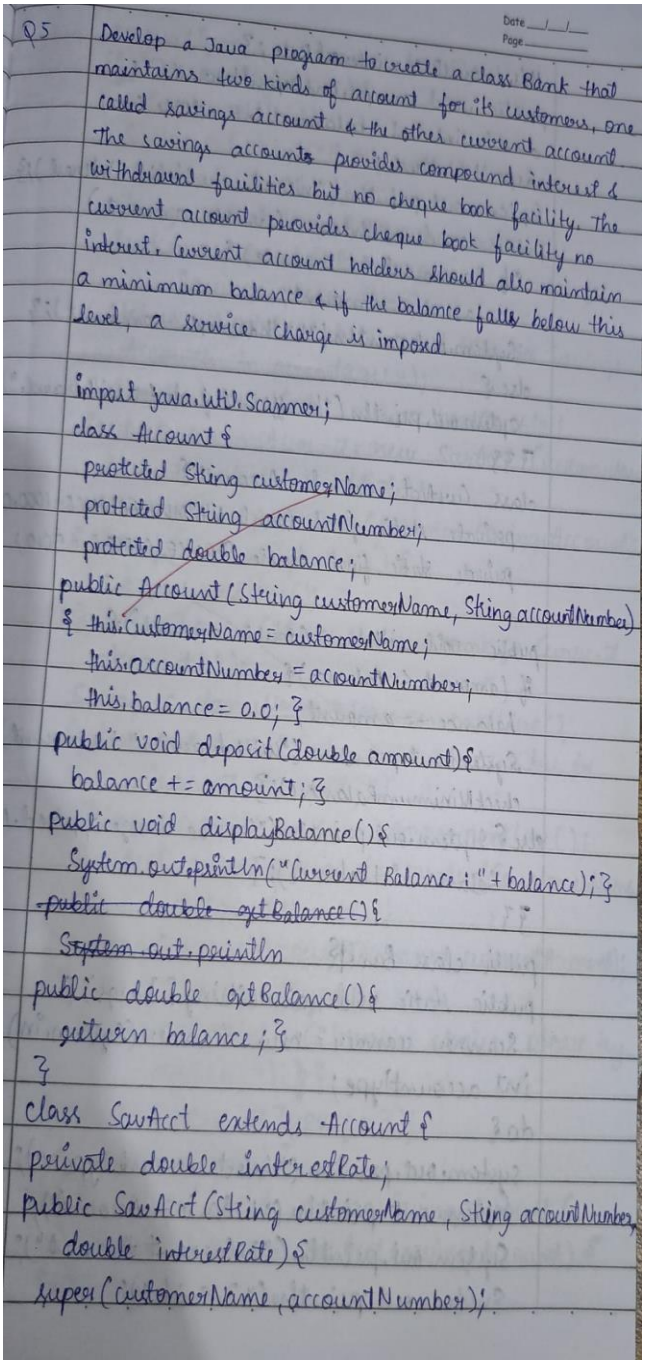
```
public class AbstractArea {
   public static void main(String[] args) {
      Shape rectangle = new Rectangle();
      rectangle.printArea();
      Shape triangle = new Triangle();
      triangle.printArea();
      Shape circle = new Circle();
      circle.printArea();
   }}
```

```
D:\1BM23CS167>javac AbstractArea.java

D:\1BM23CS167>java AbstractArea
Enter length: 3
Enter width: 4
Area of Rectangle: 12
Enter base: 3
Enter height: 4
Area of Triangle: 6.0
Enter radius: 4
Area of Circle: 50.26548245743669

D:\1BM23CS167>
```

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.

Algorithm:



```
Q5 Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account & the other current account. The savings accounts provides compound interest & withdrawal facilities but no cheque book facility. The current account provides cheque book facility no interest. Current account holders should also maintain a minimum balance & if the balance falls below this level, a service charge is imposed

import java.util.Scanner;
class Account {
protected String customerName;
protected String accountNumber;
protected double balance;
public Account(String customerName, String accountNumber)
{ this.customerName = customerName;
this.accountNumber = accountNumber;
this.balance = 0.0; }
public void deposit(double amount){
balance += amount; }
public void displayBalance(){
System.out.println("Current Balance :" + balance); }
public double getBalance(){
return balance; }
}
class SavAcct extends Account {
private double interestRate;
public SavAcct(String customerName, String accountNumber,
double interestRate){
super(customerName, accountNumber);
this.interestRate = interestRate; }
public void computeAndDepositInterest(){
double interest = balance * interestRate / 100;
deposit(interest);
System.out.println("Interest credited:" + interest); }
public void withdraw(double amount)
{ if (amount <= balance) {
balance -= amount;
System.out.println("Withdrawn:" + amount); }
else {
System.out.println("Insufficient funds for withdrawal."); }}
class CurAcct extends Account {
private static final double MINIMUM_BALANCE = 1000.0;
private static final double SERVICE_CHARGE = 50.0;

public void withdraw(double amount){
if (amount <= balance){
balance -= amount;
System.out.println("Withdrawn:" + amount);
checkMinimumBalance(); }
else { System.out.println("Service charge applied. New
Balance :" + balance); }}

public class Bank{
public static void main(String[] args){
Scanner scanner = new Scanner(System.in);
int accountType;
do{
System.out.println("Select Account Type:");
System.out.println("1. Savings Account");
System.out.println("2. Current Account");
System.out.println("3. Exit");
```

```
accountType = scanner.nextInt();
if (accountType == 1) {
  scanner.nextLine();
  System.out.print("Enter customer name for
    Savings Account: ");
  String customerName = scanner.nextLine();
  System.out.print("Enter account number for
    Savings Account: ");
  String.out.print("Enter interest rate for Savings
    Account in percentage: ");
  double interestRate = scanner.nextDouble();
  SavAcct savingsAccount = new SavAcct(customerName,
    accountNumber, interestRate);
  manageSavingsAccount(scanner, savingsAccount);
} else if (accountType == 2) {
  scanner.nextLine();
  System.out.print("Enter customer name for Current
    Account: ");
  String customerName = scanner.nextLine();
  System.out.print("Enter account number for
    Current Account: ");
  String accountNumber = scanner.nextLine();
  CurrAcct currentAccount = new CurrAcct(customerName,
    accountNumber);
  manageCurrentAccount(scanner, currentAccount);
} else if (accountType != 3) {
  System.out.println("Invalid choice! Please try
    again.");
} while (accountType != 3);
scanner.close(); }
private static void manageSavingsAccount(
  Scanner scanner, SavAcct savingsAccount) {
int choice;
```

```
do {
  System.out.println("In Saving Account operations:");
  System.out.println("1. Deposit \n 2. Withdraw
    \n 3. Compute Interest \n 4. Display Balance
    \n 5. Go Back");
  choice = scanner.nextInt();

switch (choice) {
case 1:
  System.out.print("Enter deposit amount for savings
    account: ");
  double saveDeposit = scanner.nextDouble();
  savingsAccount.deposit(savDeposit);
  break;
case 2:
  System.out.print("Enter withdraw amount from acco");
  double savWithdraw = scanner.nextDouble();
  savingsAccount.withdraw(savWithdraw);
  break;
case 3:
  savingsAccount.computeAndDepositInterest();
  break;
case 4:
  savingsAccount.displayBalance();
  break;
case 5:
  break;
default:
  System.out.println("Invalid choice!"); }
} while (choice != 5); }
```

```
Select Account Type:
1. Saving Acct
2. Current account
3. Exit
1.
Enter custom name for savings Acct: Lava
Enter acct num for savings Acct: 1345
Enter interest rate for saving acct in %: 2

Savings Acct Operations:
1. Deposit
2. Withdraw
3. Compute Interest
4. Display Balance
5. Go Back
1.
Enter deposit amt for savings Acct: 1000
Saving Acct 0.
1. Depo 2. With 3. Com. Inte 4. Disp Bal 5. Go Back
3
Interest credited: 20.0

Savings Acct operations:-
1. Deposit 2. with 3. com Int 4. Disp 5. GoBack
2
Enter withdrawal amount for saving acct: 200
withdraw: 200.0

Savings Account operations:
1. Depost 2. withdraw 3. Comp int 4. Bal 5. Go
4
Current Balance: 820.0
```

```
Saving acct operations:- 1 2 3
5.
Select Account Type:
1. Savings Account
2. Current Acct
3. Exit
2
Enter customer name for Curr acct: laxmi
Enter acct numb for Curr acct: 168

Current Account operations:
1. Deposit
2. withdraw
3. Display Balance
4. GoBack
Enter deposit amt for C.A: 1000

Current Account operati:
1. Deposit 2. Withdraw 3. Display 4. Go
2
Ente withdraw amt for C.A: 1000
withdr: 1000.0
Service charg applid. New Balance: -500

Current Acc opert
1. Dep. 2. Withd 3. Diplay 4. Goback
4
select Accid Type:
1. Saving Acc 2. Curr. Acct 3. Exit
3.
```

Code:

```
import java.util.Scanner;
class Account {
    protected String customerName;
    protected String accountNumber;
    protected double balance;
    public Account(String customerName, String accountNumber) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.balance = 0.0;
    }
    public void deposit(double amount) {
        balance += amount;
    }
    public double getBalance() {
        return balance;
    }
}
class SavAcct extends Account {
    private double interestRate;

    public SavAcct(String customerName, String accountNumber, double interestRate) {
        super(customerName, accountNumber);
        this.interestRate = interestRate;
    }
    public void computeAndDepositInterest() {
        double interest = balance * interestRate / 100;
        deposit(interest);
        System.out.println("Interest credited: " + interest);
    }
    public void displayBalance() {
        System.out.println("Current Balance: " + balance);
    }
    public void withdraw(double amount) {
        if (amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount);
        } else {
            System.out.println("Insufficient funds for withdrawal.");
        }}}
class CurAcct extends Account {
```

```
    private static final double MINIMUM_BALANCE = 1000.0;
    private static final double SERVICE_CHARGE = 50.0;
    public CurAcct(String customerName, String accountNumber) {
        super(customerName, accountNumber);
    }
    public void withdraw(double amount) {
        if (amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount);
        } else {
            System.out.println("Insufficient funds for withdrawal.");
        }}
    public void displayBalance() {
         checkMinimumBalance();
        System.out.println("Current Balance: " + balance);
    }
    private void checkMinimumBalance() {
        if (balance < MINIMUM_BALANCE) {
            balance -= SERVICE_CHARGE;
            System.out.println("Service charge applied. New Balance: " + balance);
}}}
public class Bank1 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int accountType;
        do {
            System.out.println("Select Account Type:");
            System.out.println("1. Savings Account");
            System.out.println("2. Current Account");
            System.out.println("3. Exit");
            accountType = scanner.nextInt();
            if (accountType == 1) {
                scanner.nextLine();
                System.out.print("Enter customer name for Savings Account: ");
                String customerName = scanner.nextLine();
                System.out.print("Enter account number for Savings Account: ");
                String accountNumber = scanner.nextLine();
                System.out.print("Enter interest rate for Savings Account in percentage: ");
                double interestRate = scanner.nextDouble();
                SavAcct savingsAccount = new SavAcct(customerName, accountNumber, interestRate);
                manageSavingsAccount(scanner, savingsAccount);
```

```
        } else if (accountType == 2) {
            scanner.nextLine();
            System.out.print("Enter customer name for Current Account: ");
            String customerName = scanner.nextLine();
            System.out.print("Enter account number for Current Account: ");
            String accountNumber = scanner.nextLine();
            CurAcct currentAccount = new CurAcct(customerName, accountNumber);
            manageCurrentAccount(scanner, currentAccount);
        } else if (accountType != 3) {
            System.out.println("Invalid choice! Please try again.");
        }
    } while (accountType != 3);
    scanner.close();
}
private static void manageSavingsAccount(Scanner scanner, SavAcct savingsAccount) {
    int choice;
    do {
        System.out.println("\nSavings Account Operations:");
        System.out.println("1. Deposit");
        System.out.println("2. Withdraw");
        System.out.println("3. Compute Interest");
        System.out.println("4. Display Balance");
        System.out.println("5. Go Back");
        choice = scanner.nextInt();
        switch (choice) {
            case 1:
                System.out.print("Enter deposit amount for Savings Account: ");
                double savDeposit = scanner.nextDouble();
                savingsAccount.deposit(savDeposit);
                break;
            case 2:
                System.out.print("Enter withdrawal amount for Savings Account: ");
                double savWithdraw = scanner.nextDouble();
                savingsAccount.withdraw(savWithdraw);
                break;
            case 3:
                savingsAccount.computeAndDepositInterest();
                break;
            case 4:
                savingsAccount.displayBalance();
                break;
```

```
            case 5:
                break;
            default:
                System.out.println("Invalid choice! Please try again.");
        }
    } while (choice != 5);
}
private static void manageCurrentAccount(Scanner scanner, CurAcct currentAccount) {
    int choice;
    do {
        System.out.println("\nCurrent Account Operations:");
        System.out.println("1. Deposit");
        System.out.println("2. Withdraw");
        System.out.println("3. Display Balance");
        System.out.println("4. Go Back");
        choice = scanner.nextInt();
        switch (choice) {
            case 1:
                System.out.print("Enter deposit amount for Current Account: ");
                double curDeposit = scanner.nextDouble();
                currentAccount.deposit(curDeposit);
                break;
            case 2:
                System.out.print("Enter withdrawal amount for Current Account: ");
                double curWithdraw = scanner.nextDouble();
                currentAccount.withdraw(curWithdraw);
                break;
            case 3:
                currentAccount.displayBalance();
                break;
            case 4:
                break;
            default:
                System.out.println("Invalid choice! Please try again.");
        }
} while (choice != 4);
}}
```

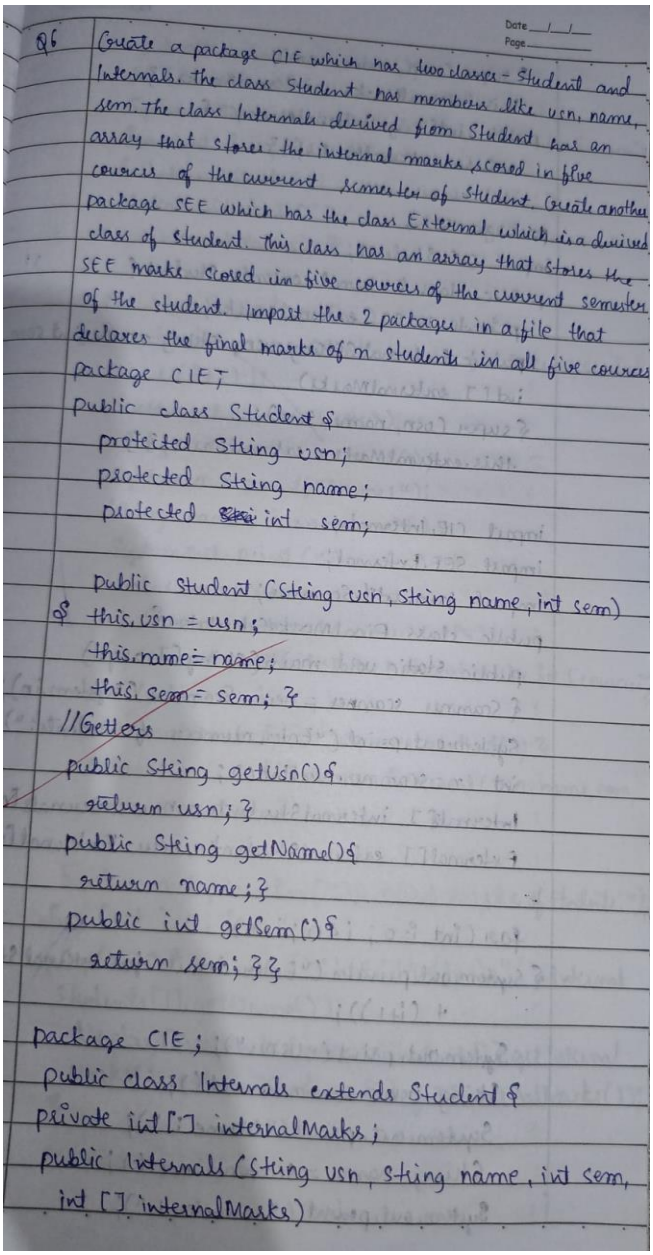
```
D:\1BM23CS167>javac Bank1.java

D:\1BM23CS167>java Bank1
Select Account Type:
1. Savings Account
2. Current Account
3. Exit
1
Enter customer name for Savings Account: lava
Enter account number for Savings Account: 1345
Enter interest rate for Savings Account in percentage: 2

Savings Account Operations:
1. Deposit
2. Withdraw
3. Compute Interest
4. Display Balance
5. Go Back
1
Enter deposit amount for Savings Account: 1000

Savings Account Operations:
1. Deposit
2. Withdraw
3. Compute Interest
4. Display Balance
5. Go Back
3
Interest credited: 20.0

Savings Account Operations:
1. Deposit
2. Withdraw
3. Compute Interest
4. Display Balance
5. Go Back
2
Enter withdrawal amount for Savings Account: 200
Withdrawn: 200.0

Savings Account Operations:
1. Deposit
2. Withdraw
3. Compute Interest
4. Display Balance
5. Go Back
4
Current Balance: 820.0
```

```
1. Deposit
2. Withdraw
3. Compute Interest
4. Display Balance
5. Go Back
5
Select Account Type:
1. Savings Account
2. Current Account
3. Exit
2
Enter customer name for Current Account: laxmi
Enter account number for Current Account: 168

Current Account Operations:
1. Deposit
2. Withdraw
3. Display Balance
4. Go Back
1
Enter deposit amount for Current Account: 1000

Current Account Operations:
1. Deposit
2. Withdraw
3. Display Balance
4. Go Back
2
Enter withdrawal amount for Current Account: 1000
Withdrawn: 1000.0
Service charge applied. New Balance: -50.0

Current Account Operations:
1. Deposit
2. Withdraw
3. Display Balance
4. Go Back
3
Service charge applied. New Balance: -100.0
Current Balance: -100.0

Current Account Operations:
1. Deposit
2. Withdraw
3. Display Balance
4. Go Back
4
Select Account Type:
1. Savings Account
2. Current Account
3. Exit
```

Program 6

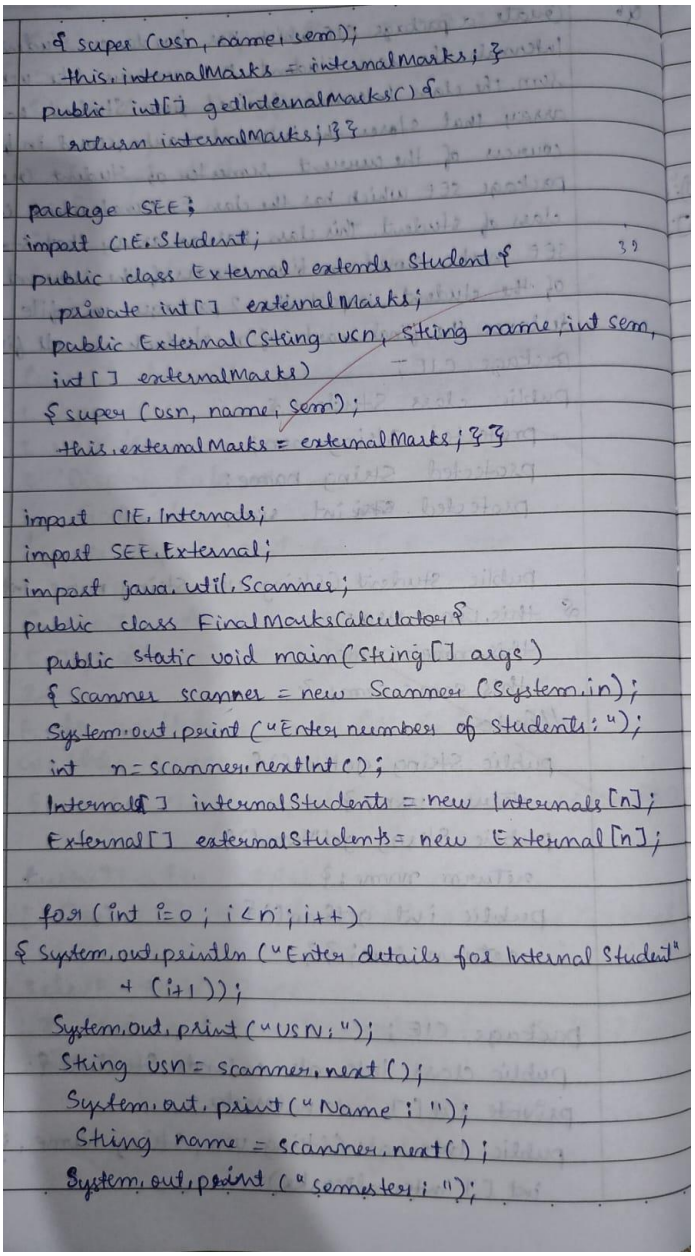
Create a package CIE which has two classes- Student and Internals. The class Student has members like usn, name, sem. The class Internals derived from Student has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has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

Algorithm:



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

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

  public Student(String usn, String name, int sem)
  { this.usn = usn;
    this.name = name;
    this.sem = sem; }
  //Getters
  public String getUsn(){
    return usn;}
  public String getName(){
    return name;}
  public int getSem(){
    return sem;}}

package CIE;
public class Internals extends Student {
private int[] internalMarks;
public Internals(String usn, String name, int sem,
  int[] internalMarks)
```



```
  { super(usn, name, sem);
    this.internalMarks = internalMarks; }
  public int[] getInternalMarks(){
    return internalMarks;}}

package SEE;
import CIE.Student;
public class External extends Student {
  private int[] externalMarks;
  public External(String usn, String name, int sem,
  int[] externalMarks)
  { super(usn, name, sem);
    this.externalMarks = externalMarks;}}

import CIE.Internals;
import SEE.External;
import java.util.Scanner;
public class FinalMarksCalculator {
  public static void main(String[] args)
  { Scanner scanner = new Scanner(System.in);
  System.out.print("Enter neumber of students: ");
  int n = scanner.nextInt();
  Internals[] internalStudents = new Internals[n];
  External[] externalStudents = new External[n];

  for(int i=0; i<n; i++)
{ System.out.println("Enter details for Internal Student"
      + (i+1));
  System.out.print("USN: ");
  String usn = scanner.next();
  System.out.print("Name: ");
  String name = scanner.next();
  System.out.print("semester: ");
```

```
int sem = scanner.nextInt();
int[] internalMarks = new int[5];
System.out.println("Enter Internal Marks for 5 courses");
for (int j=0; j<5; j++)
{ internalMarks[j] = scanner.nextInt(); }
internalStudents[i] = new Internals(usn, name, sem,
    internalMarks); }

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

System.out.println("\n Final Marks of Students:");
for (int i=0; i<n; i++)
{ System.out.println("Student "+(i+1)+":"+internal
  Students[i].getName());
  displayFinalMarks(internalStudents[i].getInternal
  -Marks(), externalStudents[i].getExternalMarks());
}
scanner.close(); }
```

```
private static void displayFinalMarks(int[]
internalMarks, int[] externalMarks){
System.out.print("Internal Marks: ");
for (int mark : internalMarks) {
  System.out.print(mark + " "); }
  System.out.print("\nExternal Marks(scaled to 50): ");
int[] scaledExternalMarks = new int[externalMarks.length];
for (int i=0; i<externalMarks.length; i++){
scaledExternalMarks[i] = (externalMarks[i]*50)/100;
System.out.print(scaledExternalMarks[i]+" "); }
System.out.print("\nTotal Marks: ");
for (int i=0; i<internalMarks.length; i++)
{ int total = internalMarks[i]+scaledExternalMarks[i];
System.out.print(total+" "); }
  System.out.println(); }}
```

Output:-

```
Enter number of students: 2
Enter details for Internal Student 1
USN: 167
Name: lab
Semester: 3
External Internal Marks for 5 courses:
45 42 47 46 41
Enter details for Internal Student 2
USN: 168
Name: Lass
Semester: 2
Enter Internal Marks for 5 courses:
45 48 47 42 41
Enter details for External Student 1
USN: 167
Name: lab
Semester: 3
Enter External Marks for 5 courses:
89 86 87 95 86
Enter details for External student 2
USN: 168
Name: Lass
Semester: 2
Enter External Marks for 5 courses:
89 87 86 87 85

Final Marks of Students:
Student 1: lab
Internal Marks: 45 42 47 46 41
External Marks (scaled to 50): 44 43 43 47 43
Total Marks: 89 85 90 93 84
Student 2: Lass
Internal Marks: 45 48 47 42 41
External Marks: 44 43 43 43 42
Total Marks: 89 91 90 85 83
```

12/4/24

Code:

```
package CIE;
public class Student {
    protected String usn;
    protected String name;
    protected int sem;
    public Student(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }
    public String getUsn() {
        return usn;
    }
    public String getName() {
        return name;
    }
    public int getSem() {
        return sem;
    }}
package CIE;
public class Internals extends Student {
    private int[] internalMarks;

    public Internals(String usn, String name, int sem, int[] internalMarks) {
        super(usn, name, sem);
        this.internalMarks = internalMarks;
    }
    public int[] getInternalMarks() {
        return internalMarks;
    }}
package SEE;
import CIE.Student;
public class External extends Student {
    private int[] externalMarks;
    public External(String usn, String name, int sem, int[] externalMarks) {
        super(usn, name, sem);
        this.externalMarks = externalMarks;
    }
    public int[] getExternalMarks() {
        return externalMarks;}}
```

```
import CIE.Internals;
import SEE.External;
import java.util.Scanner;
public class FinalMarksCalculator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of students: ");
        int n = scanner.nextInt();
        Internals[] internalStudents = new Internals[n];
        External[] externalStudents = new External[n];
        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for Internal Student " + (i + 1));
            System.out.print("USN: ");
            String usn = scanner.next();
            System.out.print("Name: ");
            String name = scanner.next();
            System.out.print("Semester: ");
            int sem = scanner.nextInt();
            int[] internalMarks = new int[5];
            System.out.println("Enter Internal Marks for 5 courses:");
            for (int j = 0; j < 5; j++) {
                internalMarks[j] = scanner.nextInt();
            }
            internalStudents[i] = new Internals(usn, name, sem, internalMarks);
        }
        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for External Student " + (i + 1));
            System.out.print("USN: ");
            String usn = scanner.next();
            System.out.print("Name: ");
            String name = scanner.next();
            System.out.print("Semester: ");
            int sem = scanner.nextInt();
            int[] externalMarks = new int[5];
            System.out.println("Enter External Marks for 5 courses:");
            for (int j = 0; j < 5; j++) {
                externalMarks[j] = scanner.nextInt();
            }
            externalStudents[i] = new External(usn, name, sem, externalMarks);
        }
        System.out.println("\nFinal Marks of Students:");
```

```
        for (int i = 0; i < n; i++) {
            System.out.println("Student " + (i + 1) + ": " + internalStudents[i].getName());
            displayFinalMarks(internalStudents[i].getInternalMarks(),
externalStudents[i].getExternalMarks());
        }
        scanner.close();
    }
    private static void displayFinalMarks(int[] internalMarks, int[] externalMarks) {
        System.out.print("Internal Marks: ");
        for (int mark : internalMarks) {
            System.out.print(mark + " ");
        }
        System.out.print("\nExternal Marks (scaled to 50): ");
        int[] scaledExternalMarks = new int[externalMarks.length];
        for (int i = 0; i < externalMarks.length; i++) {
            scaledExternalMarks[i] = (externalMarks[i] * 50) / 100;
            System.out.print(scaledExternalMarks[i] + " ");
        }
        System.out.print("\nTotal Marks: ");
        for (int i = 0; i < internalMarks.length; i++) {
            int total = internalMarks[i] + scaledExternalMarks[i];
            System.out.print(total + " ");
        }
        System.out.println();
    }}
```

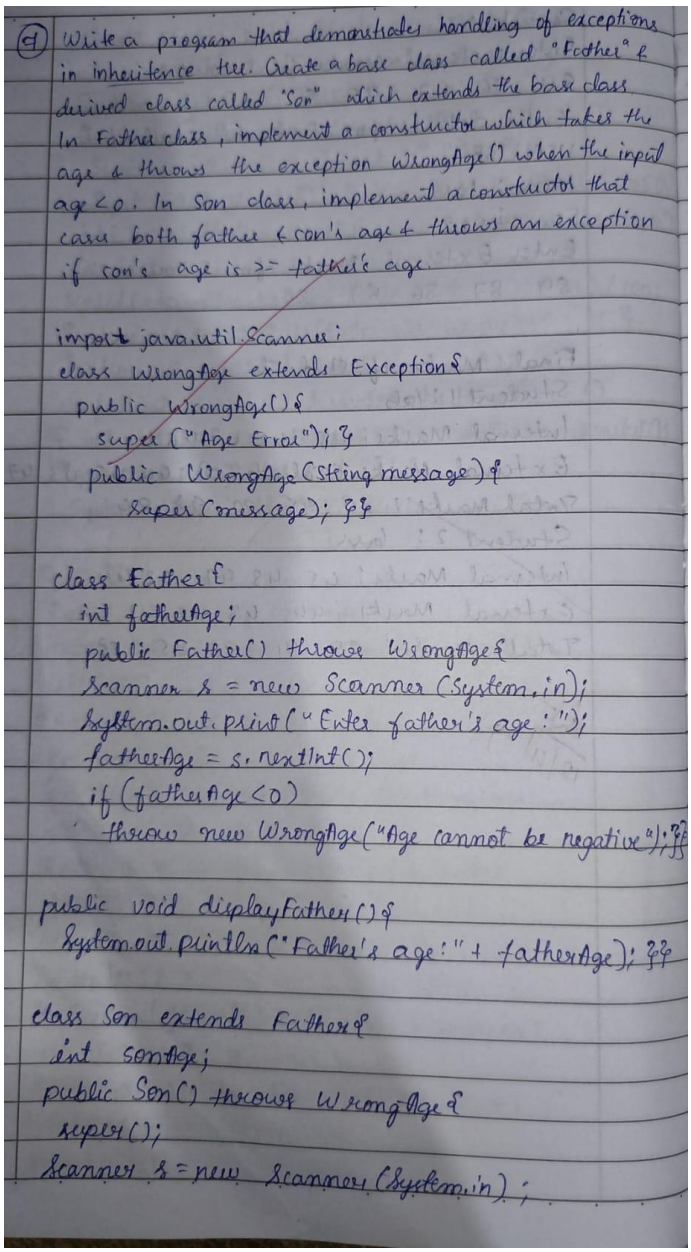
```
D:\>cd 1BM23CS167

D:\1BM23CS167>javac CIE\Student.java CIE\Internals.java SEE\External.java FinalMarksCalculator.java

D:\1BM23CS167>java FinalMarksCalculator
Enter number of students: 2
Enter details for Internal Student 1
USN: 167
Name: lab
Semester: 3
Enter Internal Marks for 5 courses:
45
42
47
46
41
Enter details for Internal Student 2
USN: 168
Name: lass
Semester: 2
Enter Internal Marks for 5 courses:
45
48
47
42
41
Enter details for External Student 1
USN: 167
Name: lab
Semester: 3
Enter External Marks for 5 courses:
89
86
87
95
86
Enter details for External Student 2
USN: 168
Name: lass
Semester: 2
Enter External Marks for 5 courses:
89
87
86
87
85
```

```
Final Marks of Students:
Student 1: lab
Internal Marks: 45 42 47 46 41
External Marks (scaled to 50): 44 43 43 47 43
Total Marks: 89 85 90 93 84
Student 2: lass
Internal Marks: 45 48 47 42 41
External Marks (scaled to 50): 44 43 43 43 42
Total Marks: 89 91 90 85 83
```

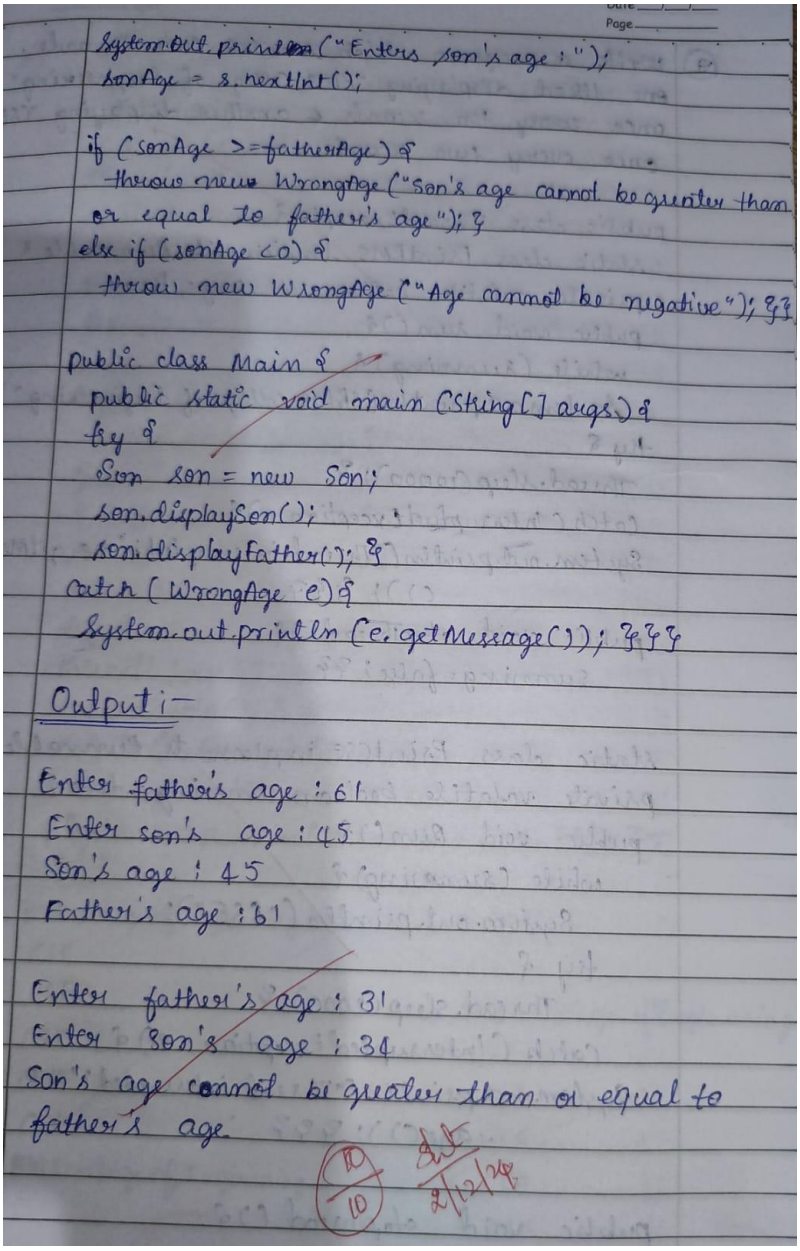
Program 7

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 cases both father and son’s age and throws an exception if son’s age is >=father’s age.

Algorithm:



```
(4) Write a program that demonstrates handling of exceptions
in inheritence tree. Create a base class called "Father" &
derived class called "Son" which extends the base class.
In Father class, implement a constructor which takes the
age & throws the exception WrongAge() when the input
age < 0. In Son class, implement a constructor that
cases both father & son's age & throws an exception
if son's age is >= father's age.

import java.util.Scanner;
class WrongAge extends Exception{
  public WrongAge(){
    super("Age Error");}
  public WrongAge(String message){
    super(message);}}

class Father{
  int fatherAge;
  public Father() throws WrongAge{
  Scanner s = new Scanner(System.in);
  System.out.print("Enter father's age:");
  fatherAge = s.nextInt();
  if(fatherAge<0)
    throw new WrongAge("Age cannot be negative");}

public void displayFather(){
  System.out.println("Father's age:"+fatherAge);}

class Son extends Father{
  int sonAge;
  public Son() throws WrongAge{
  super();
  Scanner s = new Scanner(System.in);
```



```
  System.out.print("Enter son's age:");
  sonAge = s.nextInt();

  if(sonAge >= fatherAge){
    throw new WrongAge("Son's age cannot be greater than
    or equal to father's age");}
  else if(sonAge<0){
    throw new WrongAge("Age cannot be negative");}}

public class Main{
  public static void main(String[] args){
  try{
    Son son = new Son;
    son.displaySon();
    son.displayFather();}
  catch(WrongAge e){
    System.out.println(e.getMessage());}}}
```

Output :-

```
Enter father's age : 61
Enter son's age : 45
Son's age : 45
Father's age : 61

Enter father's age : 31
Enter son's age : 34
Son's age cannot be greater than or equal to
father's age.
```

Code:

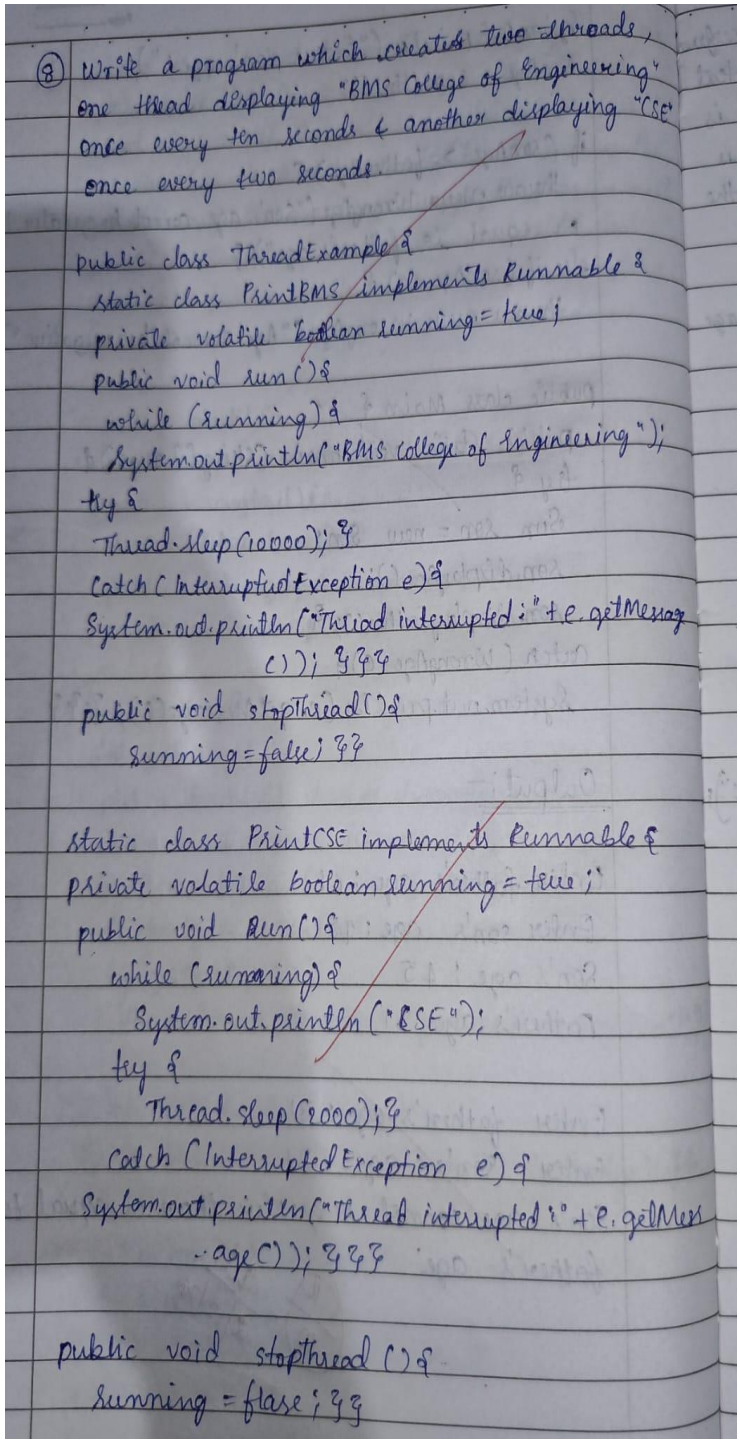
```
import java.util.Scanner;
class WrongAge extends Exception {
   public WrongAge() {
      super("Age Error");
   }
   public WrongAge(String message) {
      super(message);
   }}
class Father {
   int fatherAge;
   public Father() throws WrongAge {
      Scanner s = new Scanner(System.in);
    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

```
D:\1BM23CS167>javac Main.java

D:\1BM23CS167>java Main
Enter father's age: 61
Enter son's age: 45
Son's age: 45
Father's age: 61

D:\1BM23CS167>javac Main.java

D:\1BM23CS167>java Main
Enter father's age: 31
Enter son's age: 34
Son's age cannot be greater than or equal to father's age

D:\1BM23CS167>
```

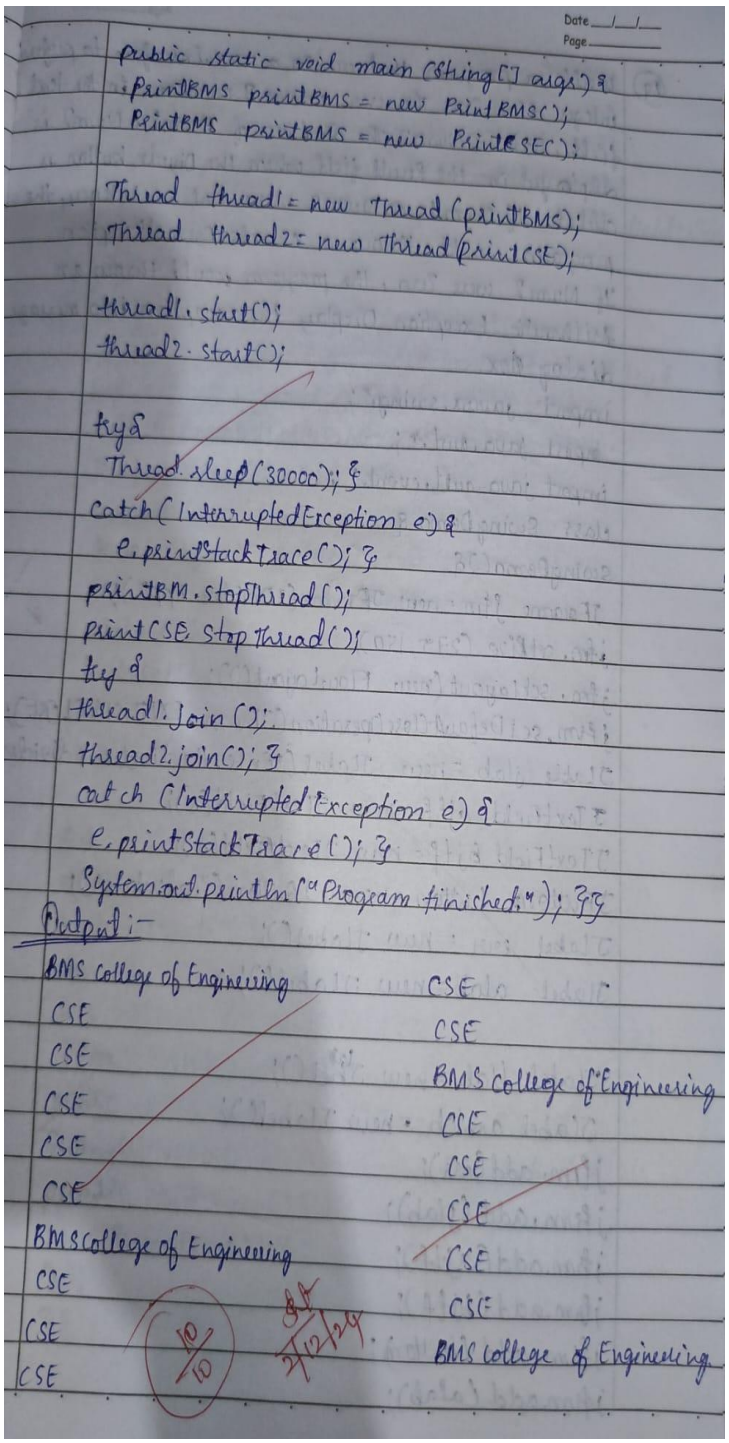
Program 8

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:



```
⑧ Write a program which creates two threads,
one thread displaying "BMS College of Engineering"
once every ten seconds & another displaying "CSE"
once every two seconds.

public class ThreadExample {
 static class PrintBMS implements Runnable {
 private volatile boolean running = true;
 public void run() {
  while (running) {
  System.out.println("BMS College of Engineering");
 try {
 Thread.sleep(10000); }
 Catch (InterruptedException e) {
 System.out.println("Thread interrupted:" + e.getMessag
                ()); }}}
 public void stopThread() {
    running = false; }}

 static class PrintCSE implements Runnable {
 private volatile boolean running = true;
 public void Run() {
   while (running) {
    System.out.println("CSE");
   try {
     Thread.sleep(2000); }
    Catch (InterruptedException e) {
 System.out.println("Thread interrupted:" + e.getMes
        -sage()); }}}

 public void stopthread() {
   running = flase; }}
```



```
public static void main (String[] args) {
 PrintBMS printBMS = new PrintBMS();
 PrintBMS printBMS = new PrintCSE();

Thread thread1 = new Thread(printBMS);
Thread thread2 = new Thread(printCSE);

thread1.start();
thread2.start();

try {
 Thread.sleep(30000); }
catch (InterruptedException e) {
 e.printStackTrace(); }
psrintBM.stopthread();
printCSE.stopThread();
try {
thread1.join();
thread2.join(); }
catch (InterruptedException e) {
 e.printStackTrace(); }
 System.out.println("Program finished."); }}
```

Output:-

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

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

Code:

```
public class ThreadExample {
    static class PrintBMS implements Runnable {
        private volatile boolean running = true;
        public void run() {
            while (running) {
                System.out.println("BMS College of Engineering");
                try {
                    Thread.sleep(10000); // Sleep for 10 seconds
                } catch (InterruptedException e) {
                    System.out.println("Thread interrupted: " + e.getMessage());
                }}}
        public void stopThread() {
            running = false;
        }}
    static class PrintCSE implements Runnable {
        private volatile boolean running = true;
        public void run() {
            while (running) {
                System.out.println("CSE");
                try {
                    Thread.sleep(2000);
                } catch (InterruptedException e) {
                    System.out.println("Thread interrupted: " + e.getMessage());
                }}}
        public void stopThread() {
            running = false;
        }}
    public static void main(String[] args) {
        PrintBMS printBMS = new PrintBMS();
        PrintCSE printCSE = new PrintCSE();
        Thread thread1 = new Thread(printBMS);
        Thread thread2 = new Thread(printCSE);
        thread1.start();
        thread2.start();
        try {
            Thread.sleep(30000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        printBMS.stopThread();
        printCSE.stopThread();
        try {
            thread1.join();
            thread2.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
```

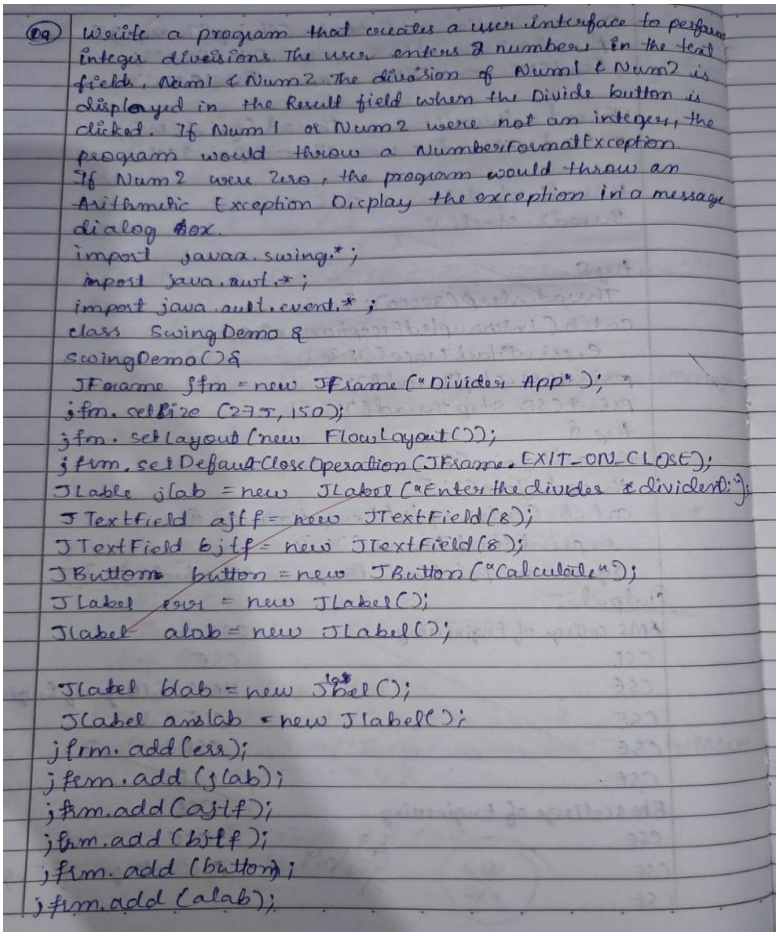
```
    }
    System.out.println("Program finished.");
}}
```

```
D:\1BM23CS167>javac ThreadExample.java

D:\1BM23CS167>java ThreadExample
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
Program finished.

D:\1BM23CS167>
```

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.

Algorithm



Write a program that creates a user interface to perform integer divisions. The user enters 2 numbers in the text fields, Num1 & Num2. The division of Num1 & 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 javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class SwingDemo {
SwingDemo() {
JFrame jfm = new JFrame("Divider App");
jfm.setSize(275, 150);
jfm.setLayout(new FlowLayout());
jfm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JLable jlab = new JLabel("Enter the divider & dividend:");
JTextField ajtf = new JTextField(8);
JTextField bjtf = new JTextField(8);
JButton button = new JButton("Calculate");
JLabel err = new JLabel();
JLabel alab = new JLabel();

JLabel blab = new JLabel();
JLabel anslab = new JLabel();
jfm.add(err);
jfm.add(jlab);
jfm.add(ajtf);
jfm.add(bjtf);
jfm.add(button);
jfm.add(alab);
```

```
jfm.add(blab);
jfm.add(anslab);
ActionListener l = new ActionListener() {
public void actionPerformed(ActionEvent evt) {
System.out.println("Action event from a text field");}};
ajtf.addActionListener(l);
bjtf.addActionListener(l);
button.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent evt) { try {
int a = Integer.parseInt(ajtf.getText()); int b =
Integer.parseInt(bjtf.getText()); int ans = a/b;
alab.setText("\n A = " + a);
blab.setText("\n B = " + b);
anslab.setText("\n Ans = " + ans); }
catch (NumberFormatException e) {
alab.setText("");
blab.setText("");
anslab.setText("");
err.setText("Enter only Integers!"); }

catch(ArithmeticException e) {
alab.setText("");
blab.setText("");
anslab.setText("");
err.setText("B should be NON zero!"); }});
jfm.setVisible(true); }
public static void main(String args[]) {
SwingUtilities.invokeLater(new Runnable() {
public void run() {
new SwingDemo(); }
});
}}
```

Output :-

Divider App
B should be Non zero!
Enter the divider & dividend
[2] [0] [calculate]

Divider App
Enter only integers!
Enter the divider & dividend
[2] [a] [calculate]

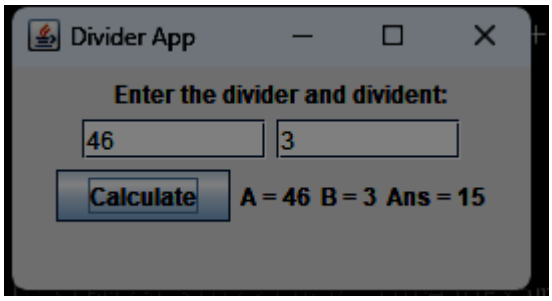
2/12/24

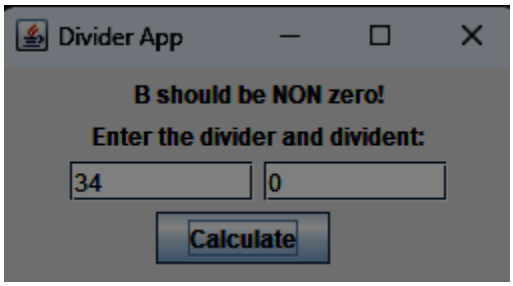
10/10

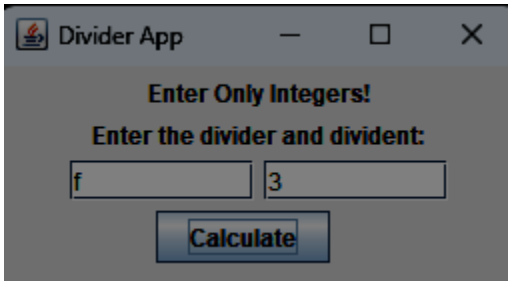
Code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class SwingDemo{
SwingDemo(){
JFrame jfrm = new JFrame("Divider App");
jfrm.setSize(275, 150);
jfrm.setLayout(new FlowLayout());
jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JLabel jlab = new JLabel("Enter the divider and divident:");
// add text field for both numbers
JTextField ajtf = new JTextField(8);
JTextField bjtf = new JTextField(8);
// calc button
JButton button = new JButton("Calculate");
JLabel err = new JLabel();
JLabel alab = new JLabel();

JLabel blab = new JLabel();
JLabel anslab = new JLabel();
jfrm.add(err);
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);
ActionListener l = new ActionListener() {
public void actionPerformed(ActionEvent evt) {
System.out.println("Action event from a text field"); }
};
ajtf.addActionListener(l);
bjtf.addActionListener(l);
button.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent evt) { try{
int a = Integer.parseInt(ajtf.getText()); int b =
Integer.parseInt(bjtf.getText()); int ans = a/b;
alab.setText("\nA = " + a);
blab.setText("\nB = " + b);
anslab.setText("\nAns = "+ ans);
}
catch(NumberFormatException e){
alab.setText("");
blab.setText("");
anslab.setText("");
```

```
err.setText("Enter Only Integers!"); }

catch(ArithmeticException e){
alab.setText("");
blab.setText("");
anslab.setText("");
err.setText("B should be NON zero!"); }
}
});
// display frame
jfrm.setVisible(true);
}
public static void main(String args[]){ // create
SwingUtilities.invokeLater(new Runnable(){
public void run(){
new SwingDemo();
}
});
}
}
```



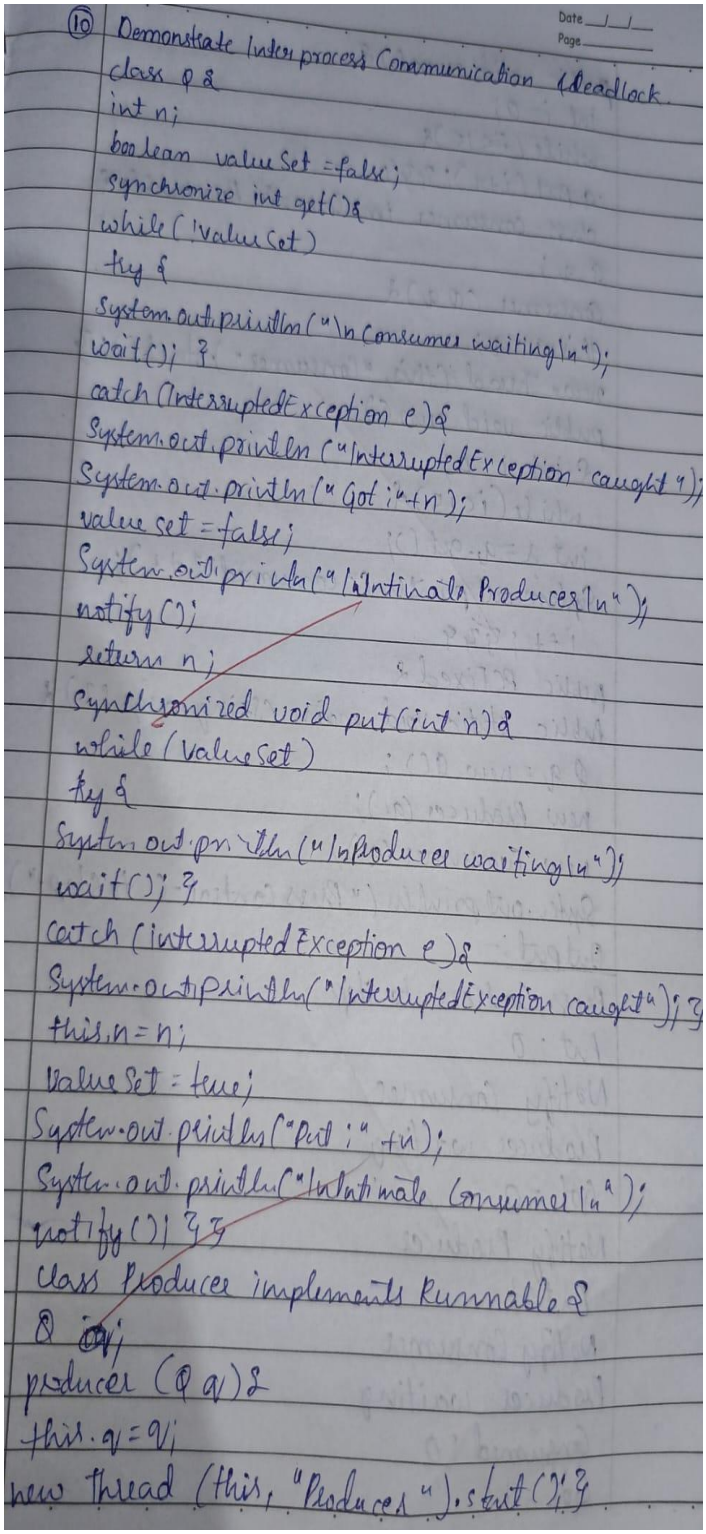




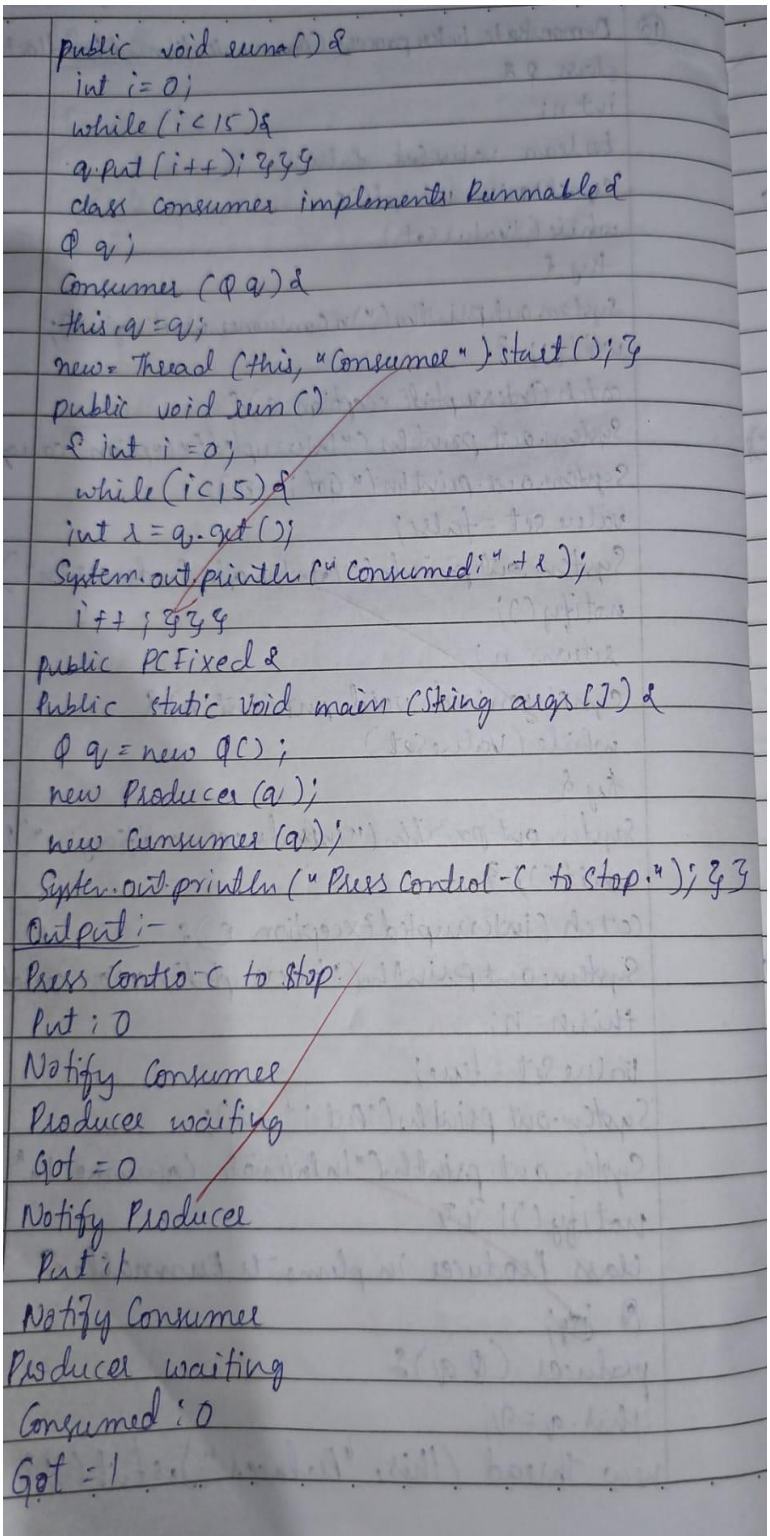
Program 10

Demonstrate Inter process Communication and deadlock

Algorithm:



```
(10) Demonstrate Inter process Communication & Deadlock
class Q {
int n;
boolean valueSet = false;
synchronize int get(){
while(!valueSet)
try {
System.out.println("\n Consumer waiting\n");
wait(); }
catch (InterruptedException e){
System.out.println("InterruptedException caught");
System.out.println("Got:"+n);
valueSet = false;
System.out.println("\n Intimate Producer\n");
notify();
return n;
synchronized void put(int n){
while(valueSet)
try {
System.out.println("\n Producer waiting\n");
wait(); }
catch (InterruptedException e){
System.out.println("\n InterruptedException caught"); }
this.n = n;
valueSet = true;
System.out.println("Put:" +n);
System.out.println("\n Intimate 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++;}}}
public 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:-

```
Press Control-C to stop
Put : 0
Notify Consumer
Producer waiting
Got = 0
Notify Producer
Put : 1
Notify Consumer
Producer waiting
Consumed : 0
Got = 1
```

Notify Producer
consumed :)
put : 2
Notify Consumer
produces waiting
got : 2
Notify Producer
consumed : 2 .

10/10

Sk
2/12/24

Deadlock

```
class A {
synchronized void foo(B b) {
String name = Thread.currentThread().getName();
System.out.println(name + "entered A.foo");
try {
Thread.sleep(1000); }
catch(Exception e){
System.out.println("A interrupted"); }
System.out.println(name + "trying to call B.last()");
b.last(); }
void last() {
System.out.println("inside A.last"); } }
class B {
Synchronized void bar(A a) {
String name = Thread.currentThread().getName();
System.out.println(name + "entered B.bar");
try {
Thread.sleep(1000); }
catch(Exception e){
System.out.println("B interrupted");
System.out.println(name + "trying to call A.last()");
a.last(); }
void last() {
System.out.println("Inside A.last"); } }
class Deadlock implements Runnable {
A a = new A();
B b = new B();
Deadlock() {
public void run() {
b.bar(a);
thread)
```

```
System.out.println("Back in other thread"); }
public static void main(String args[]) {
new Deadlock(); } }
```

Output:-

Mainthread entered A.foo
Racingthread entered B.bar
Mainthread trying to call B.last()
Inside A.last
Back in main thread
Racingthread trying to call A.last()
Inside A.last
Back in otherthread.

10/10

Sk
2/12/24

Code:

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

```
D:\1BM23CS167>javac PCFixed.java

D:\1BM23CS167>java PCFixed.java
Press Control-C to stop.
Put: 0

Intimate Consumer


Producer waiting

Got: 0

Intimate Producer

Put: 1

Intimate Consumer


Producer waiting

consumed:0
Got: 1

Intimate Producer

consumed:1
Put: 2

Intimate Consumer


Producer waiting
```

```
Producer waiting

Got: 2

Intimate Producer

consumed:2
Put: 3

Intimate Consumer


Producer waiting

Got: 3

Intimate Producer

consumed:3
Put: 4

Intimate Consumer


Producer waiting

Got: 4

Intimate Producer

consumed:4
Put: 5
```

```
Intimate Consumer


Producer waiting

Got: 5

Intimate Producer

consumed:5
Put: 6

Intimate Consumer


Producer waiting

Got: 6

Intimate Producer

consumed:6
Put: 7

Intimate Consumer


Producer waiting

Got: 7

Intimate Producer

consumed:7
```

```
consumed:7
Put: 8

Intimate Consumer


Producer waiting

Got: 8

Intimate Producer

consumed:8
Put: 9

Intimate Consumer


Producer waiting

Got: 9

Intimate Producer

consumed:9
Put: 10

Intimate Consumer


Producer waiting

Got: 10
```

```
Intimate Producer

consumed:10
Put: 11

Intimate Consumer


Producer waiting
```

```
Got: 11

Intimate Producer

consumed:11
Put: 12

Intimate Consumer


Producer waiting
```

```
Got: 12

Intimate Producer

consumed:12
Put: 13

Intimate Consumer


Producer waiting
```

```
Got: 13

Intimate Producer

consumed:13
Put: 14

Intimate Consumer

Got: 14

Intimate Producer

consumed:14
```

Code:

```
class A {
synchronized void foo(B b) {
String name =Thread.currentThread().getName();
System.out.println(name + " entered A.foo");
try {
Thread.sleep(1000);
} catch(Exception e) {
System.out.println("A Interrupted");
}
System.out.println(name + " trying to call B.last()");
b.last();
}
void last() {
System.out.println("Inside A.last");
}}
class B {
synchronized void bar(A a) {
String name =Thread.currentThread().getName();
System.out.println(name + " entered B.bar");
try {
Thread.sleep(1000);
} catch(Exception e) {
System.out.println("B Interrupted");
}
System.out.println(name + " trying to call A.last()");
a.last();}
void last() {
System.out.println("Inside A.last");
}}
class 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
System.out.println("Back in main thread");
}
public void run() {
b.bar(a); // get lock on b in other
System.out.println("Back in other thread");
}
public static void main(String args[]) {
new Deadlock();}}
```

```
D:\1BM23CS167>javac Deadlock.java

D:\1BM23CS167>java Deadlock
RacingThread entered B.bar
MainThread entered A.foo
RacingThread trying to call A.last()
MainThread trying to call B.last()
Inside A.last
Inside A.last
Back in main thread
Back in other thread
```