VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum-590014, Karnataka.

LAB REPORT

On

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

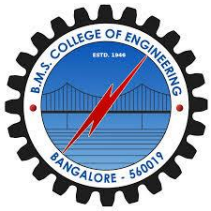
DHRUVA S RAO (1BM23CS092)

In partial fulfilment 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, Bengaluru-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(23CSPCOOJ)” carried out by **DHRUVA S RAO(1BM23CS092)**, who is a bonafide student of **BMS College of Engineering**. It is in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024. The Lab report has been approved as it satisfies the academic requirements in respect of a Object Oriented Java Programming(23CSPCOOJ) work prescribed for the said degree.

Dr. Swathi Sridharan Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor HOD Department of CSE, BMSCE

Github Link:

https://github.com/DhruvaSRao64/Java-Program

Program 1

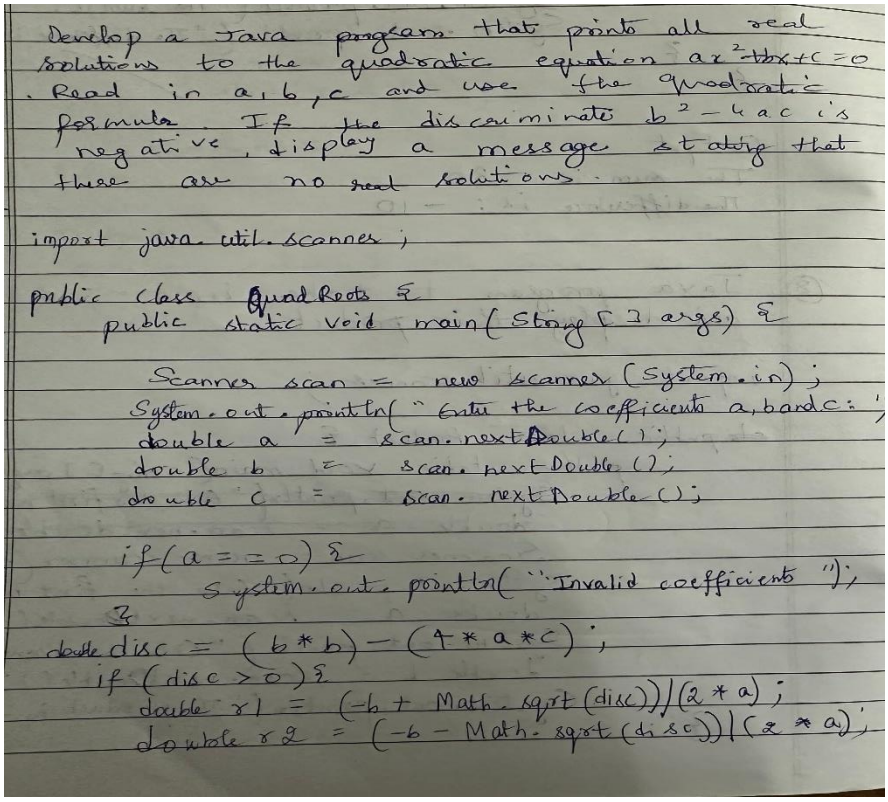
Implement Quadratic Equation

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.

Source code:

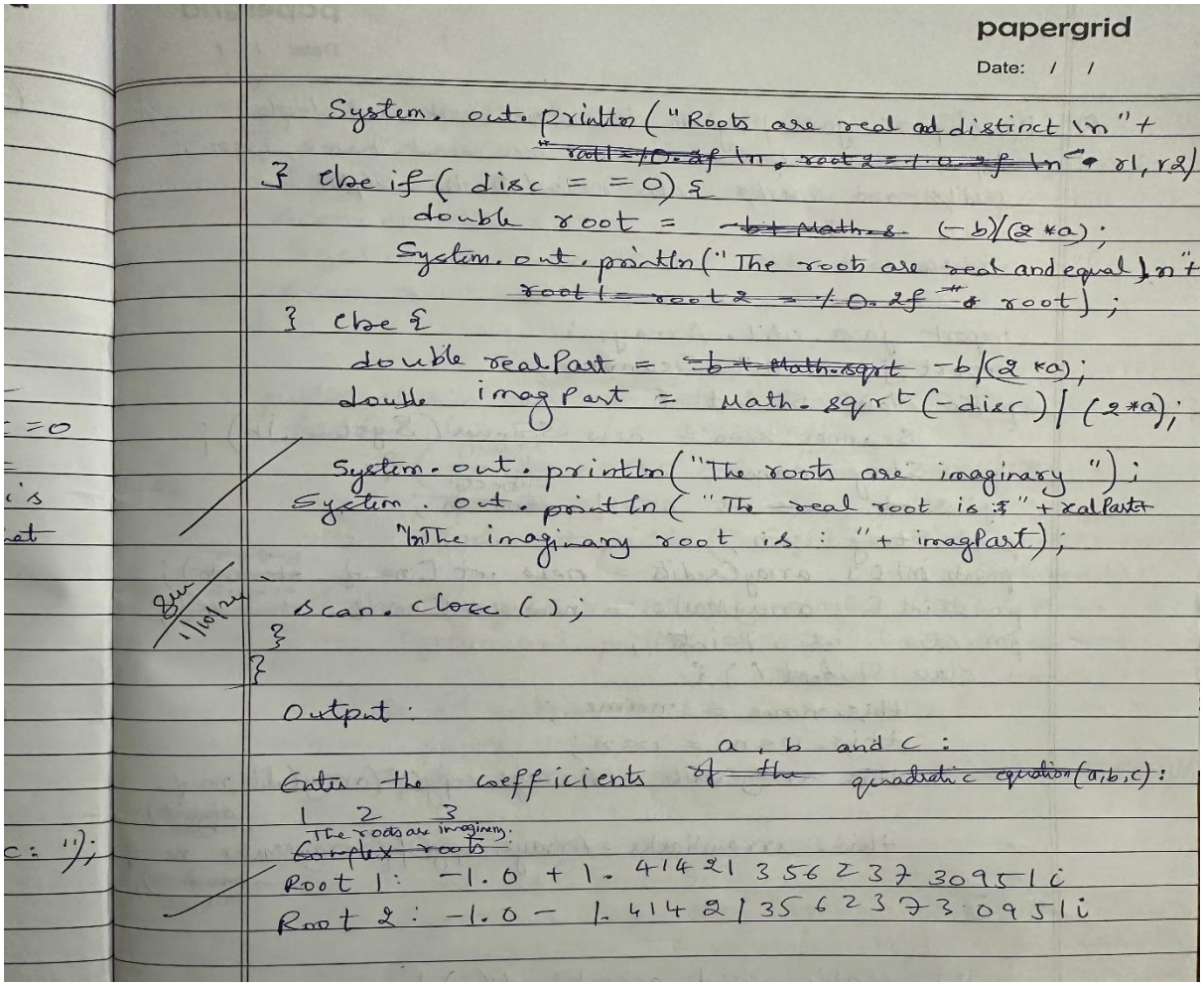
```
import java.util.Scanner;
public class Quadratic {
    public static void main(String[] args){
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the coefficient a: ");
        double a = scanner.nextDouble();
        System.out.println("Enter the coefficient b: ");
        double b = scanner.nextDouble();
        System.out.println("Enter the coefficient c: ");
        double c = scanner.nextDouble();
        if (a == 0) {
            System.out.println("Invalid coefficient of a!");
        }
        double disc = (b*b) - (4 * a * c);
        if (disc > 0) {
            double root1 = -b + (Math.sqrt(disc)/(2 * a));
            double root2 = -b - (Math.sqrt(disc)/(2 * a));
            System.out.println("Roots are real and distinct");
            System.out.println("The first root is: "+root1+"\n The second root is: "+root2);
        } else if (disc == 0) {
            double root1 = -b/ (2 * a);
            double root2 = root1;
            System.out.println("Roots are real and equal");
            System.out.println("The roots are: "+root1+"\n"+root2);
        } else {
            double realPart = -b/ (2 * a);
            double imagPart = Math.sqrt(-disc)/ (2 * a);
            System.out.println("The equation has complex roots");
            System.out.println("The real root is: "+realPart+"\nThe imaginary part is: "+imagPart);
        }
```

```
        scanner.close();
    }
}
```



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

```
import java.util.Scanner;

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

        Scanner scan = new Scanner(System.in);
        System.out.println("Enter the coefficients a, b and c: ");
        double a = scan.nextDouble();
        double b = scan.nextDouble();
        double c = scan.nextDouble();

        if(a == 0) {
            System.out.println("Invalid coefficients");
        }
        double disc = (b*b) - (4*a*c);
        if (disc > 0) {
            double r1 = (-b + Math.sqrt(disc))/(2*a);
            double r2 = (-b - Math.sqrt(disc))/(2*a);
```



papergrid

Date: / /

```
System.out.println("Roots are real and distinct \n" +
                   "root1 = %.2f \n, root2 = %.2f \n", r1, r2);
} else if (disc == 0) {
    double root = (-b)/(2*a);
    System.out.println("The roots are real and equal \n" +
                       "root1 = root2 = %.2f ", root);
} else {
    double realPart = -b/(2*a);
    double imagPart = Math.sqrt(-disc)/(2*a);

    System.out.println("The roots are imaginary");
    System.out.println("The real root is: " + realPart +
                       "\nThe imaginary root is: " + imagPart);

    scan.close();
}
}
```

Output:

```
Enter the coefficients of a, b and c:
1 2 3
The roots are imaginary.
Complex roots
Root 1: -1.0 + 1.4142135623730951i
Root 2: -1.0 - 1.4142135623730951i
```

```
dhruvasrao@Dhruvas-MacBook-Pro p % javac Quadratic.java
dhruvasrao@Dhruvas-MacBook-Pro p % java Quadratic
Enter the coefficient a:
4
Enter the coefficient b:
2
Enter the coefficient c:
1
The equation has complex roots
The real root is: -0.25
The imaginary part is: 0.4330127018922193
dhruvasrao@Dhruvas-MacBook-Pro p % java Quadratic
Enter the coefficient a:
5
Enter the coefficient b:
8
Enter the coefficient c:
2
Roots are real and distinct
The first root is: -7.510102051443364
 The second root is: -8.489897948556635
dhruvasrao@Dhruvas-MacBook-Pro p % java Quadratic
Enter the coefficient a:
2
Enter the coefficient b:
4
Enter the coefficient c:
2
Roots are real and equal
The roots are: -1.0
-1.0
dhruvasrao@Dhruvas-MacBook-Pro p %
```

Ln 36, Col 25 Spaces: 4 UTF-8 LF {} Java Go Live

OUTPUT 1

Program 2

Student's SGPA Calculator

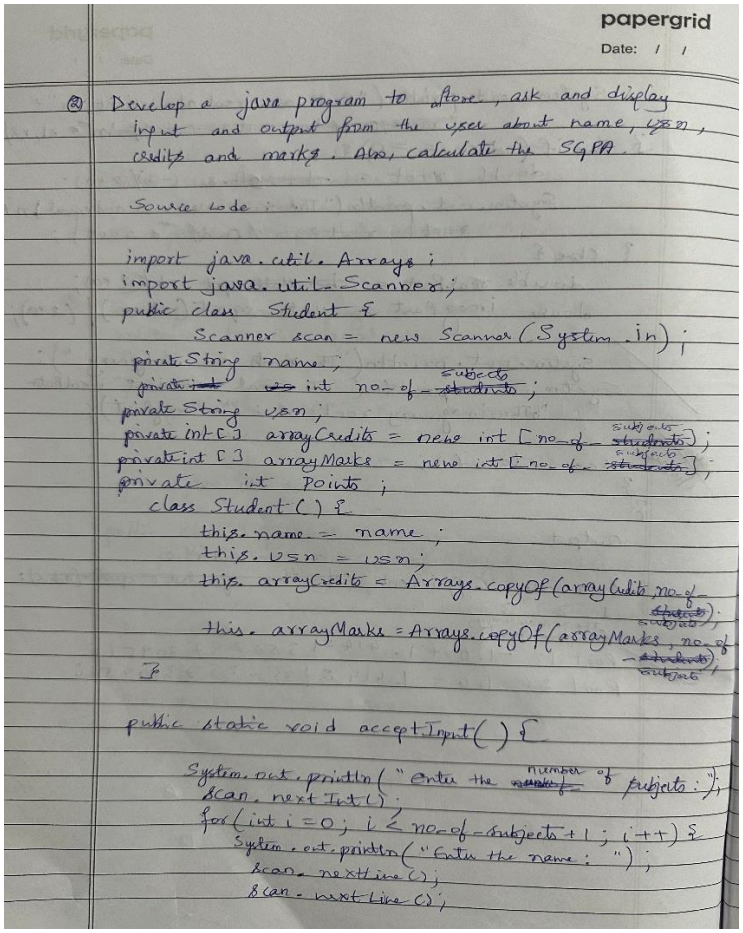
Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

Source Code:

```
import java.util.Scanner;
class Student {
    private String usn;
    private String name;
    private int [] credits;
    private int [] marks;
    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 number of subjects: ");
        int n = scanner.nextInt();
        credits = new int[n];
        marks = new int[n];
        for (int i = 0; i < n; 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();
        }
        scanner.close();
    }
    public void displayDetails() {
```

```
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Subject Details:");
        for (int i = 0; i < credits.length; i++) {
            System.out.println("Subject " + (i + 1) + " - Credits: " + credits[i] + ", Marks: " + marks[i]);
        }
    }
    public double calculateSGPA() {
        int totalCredits = 0;
        int weightedGradePoints = 0;
        for (int i = 0; i < credits.length; i++) {
            int gradePoints = getGradePoints(marks[i]);
            weightedGradePoints += gradePoints * credits[i];
            totalCredits += credits[i];
        }
        if (totalCredits == 0) {
            return 0.0;
        }
        return (double) weightedGradePoints / totalCredits;
    }
    private int getGradePoints(int marks) {
        if (marks >= 90) return 10;
        if (marks >= 80) return 9;
        if (marks >= 70) return 8;
        if (marks >= 60) return 7;
        if (marks >= 50) return 6;
        if (marks >= 40) return 5;
        return 0;
    }
}
public class StudentSGPACalculator {
    public static void main(String[] args) {
        Student student = new Student();
        student.acceptDetails();
        student.displayDetails();
        double sgpa = student.calculateSGPA();
```

```
        System.out.println("SGPA: " + sgpa);
    }
}
```



papergrid
Date: / /

② Develop a java program to store, ask and display input and output from the user about name, usn, credits and marks. Also, calculate the SGPA.

Source code:

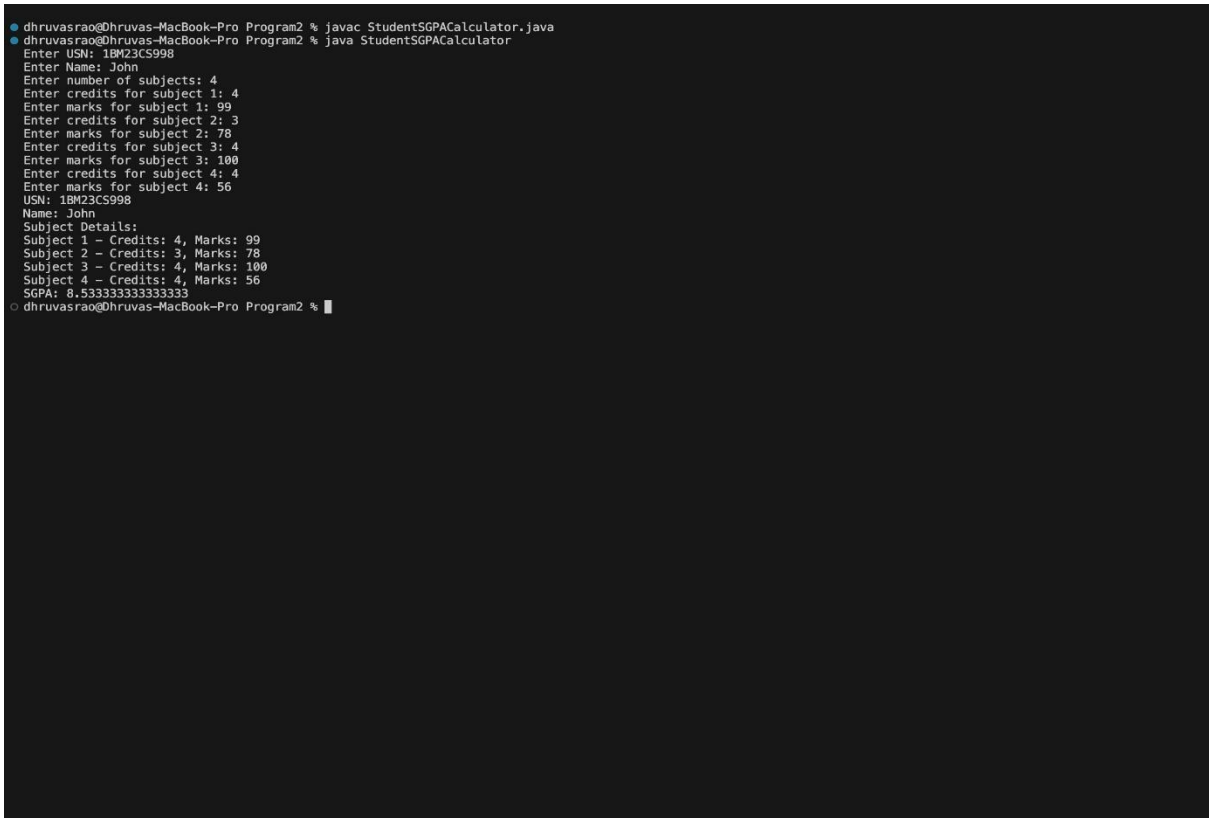
```
import java.util.Arrays;
import java.util.Scanner;
public class Student {
    Scanner scan = new Scanner(System.in);
private String name;
private int no_of_subjects;
private String usn;
private int[] arrayCredits = new int[no_of_subjects];
private int[] arrayMarks = new int[no_of_subjects];
private int Points;
  class Student() {
    this.name = name;
    this.usn = usn;
    this.arrayCredits = Arrays.copyOf(arrayCredits, no_of_subjects);
    this.arrayMarks = Arrays.copyOf(arrayMarks, no_of_subjects);
  }

public static void acceptInput() {
    System.out.println("Enter the number of subjects:");
    scan.nextInt();
    for(int i=0; i<no_of_subjects+1; i++) {
        System.out.println("Enter the name:");
        scan.nextLine();
        scan.nextLine();
```

```
System.out.println("Enter the USN : ");
scan.nextLine(); for(int i=0; i<no_of_subjects+1; i
=>System.out.println("Enter the number of credits :
scan.nextInt();
System.out.println("Enter the marks : ");
scan.nextInt();
}

public static void calculateSGPA() {
  for(int i = 0; i < arrayCredits; i++) {
    if(marks >= 90 && marks <= 100) {
      points = 10;
      System.out.println("SGPA is : "+(points * arrayCredits[i])/(arrayCredits));
    } else if(marks >= 80 && marks < 90) { points = 9
      System.out.println("SGPA is : "+(points * arrayCredits[i])/(credits));
    } else if(marks >= 70 && marks < 80) {
      points = 8;
      System.out.println("SGPA is : "+(points * arrayCredits[i])/(credits));
    } else if(marks >= 60 && marks < 70) {
      points = 7;
      System.out.println("SGPA is : "+(points * arrayCredits[i])/(credits));
    } else if(marks >= 50 && marks < 60) {
      points = 6;
      System.out.println("SGPA is : "+(points * arrayCredits[i])/(credits));
    } else {
      System.out.println("Fail");
    }
  return arrayCredits == 0 ? 0 : arrayCredits;
}
```

```
public static void display() {
  System.out.println("Name : "+name+"\nUSN : "+USN);
  System.out.println("The SGPA is : "+calculateSGPA());
}

public static void main(String[] args) {
  Student S1 = new Student(2);
  acceptInput();
  calculateSGPA();
  display();
  System.out.println("Thank You !");
}
```

Output

```
Enter the number of subjects : 2
Enter USN : 1BM23CS092
Enter name : Dhruvan
Enter credits for subject 1 : 3
Enter marks for subject 1 : 67
Enter credits for subject 2 : 4
Enter marks for subject 2 : 89
USN : 1BM23CS092
Name : a
Subject 1 : credits : 3, Marks : 67
Subject 2 : Credits : 4, Marks : 89
SGPA : 8.14
```



OUTPUT 2

Program 3

Book Information

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

Source code:

```
import java.util.Scanner;

public class Book {

    String name;

    String author;

    double price;

    int num_pages;

    public Book (String name, String author, double price, int num_pages){

        this.name = name;

        this.author = author;

        this.price = price;
```

```
        this.num_pages = num_pages;
    }
    public String getName(){
        return name;
    }
    public void setName(String name){
        this.name = name;
    }
    public String getAuthor(){
        return author;
    }
    public void setAuthor(String author){
        this.author = author;
    }
    public double getPrice(){
        return price;
    }
    public void setPrice(double price){
        this.price = price;
    }
    public int getNumPages(){
        return num_pages;
    }
    public void setNumPages(int num_pages){
        this.num_pages = num_pages;
    }
    @Override
    public String toString(){
        return "The book name is: "+name+"\nAuthor: "+author+"\nPrice: "+price+"Number of pages: "+num_pages;
        }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the number of books: ");
        int n = scanner.nextInt();
```

```
        scanner.nextLine();
        Book[] book = new Book[n];
        System.out.println("Enter the details: ");
        for (int i = 0; i < n; i++){
            System.out.println("Enter the name for "+"Book: "+(i+1));
            String name = scanner.nextLine();
            System.out.println("Enter the author name: ");
            String author = scanner.nextLine();
            System.out.println("Enter the price: ");
            double price = scanner.nextDouble();
            System.out.println("Enter the number of pages: ");
            int num_pages = scanner.nextInt();
            scanner.nextLine();
            book[i] = new Book(name, author, price, num_pages);
        }
        System.out.println("The book details is: ");
        for (Book books: book){
            System.out.println(books.toString());
        }
        scanner.close();
    }
}
```

papergrid
Date: / /

Lab Program - 3

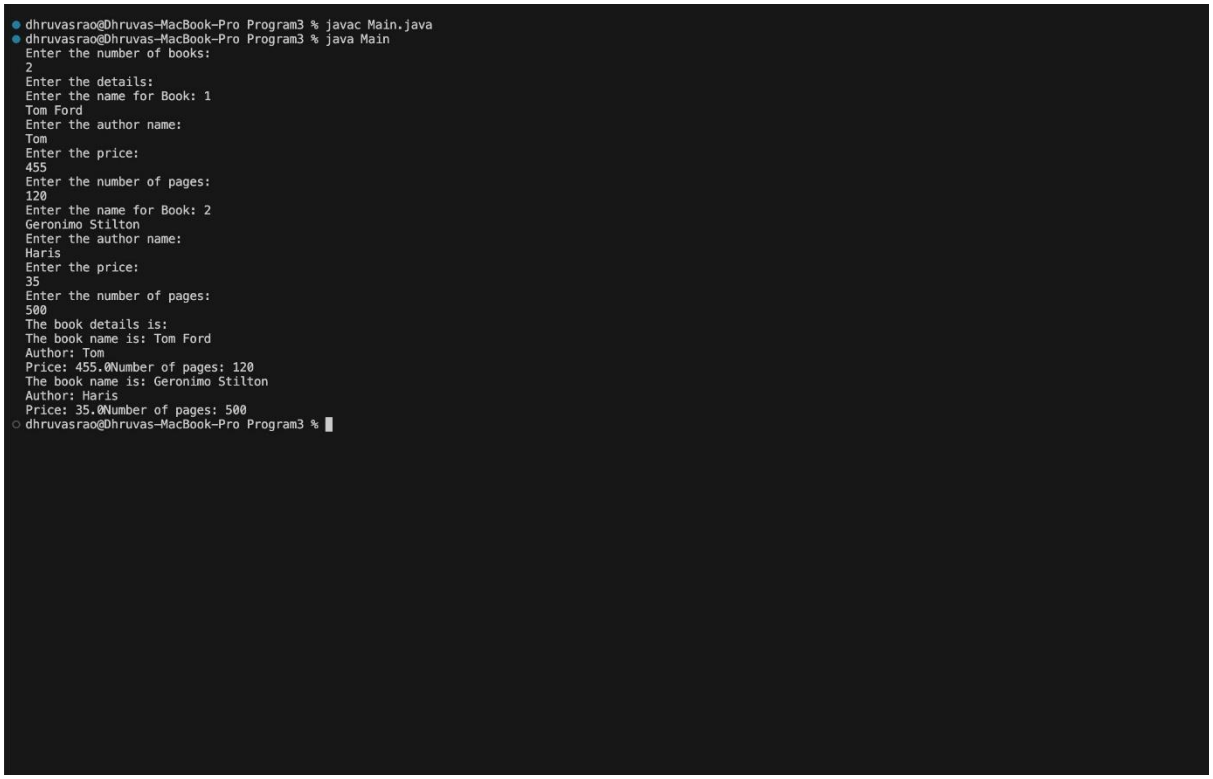
Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString() method that could display the complete details of the book. Develop a Java program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

Source Code:

```
import java.util.ArrayList;
import java.util.Scanner;

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

    public Book(String name, String author, double price, int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }
    public void setName(String name) {
        this.name = name;
    }
    public void setAuthor(String author) {
        this.author = author;
    }
    public void setPrice(double price) {
        this.price = price;
    }
    public void setNumPages(int numPages) {
        this.numPages = numPages;
    }
    public String getName() {
        return name;
    }
    public String getAuthor() {
        return author;
    }
    public double getPrice() {
        return price;
    }
    public int getNumPages() {
        return numPages;
    }
    @Override
    public String toString() {
        return "Book name = " + name + "author = " + author + "price = " + price + "numPages = " + numPages;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        ArrayList<Book> books = new ArrayList<>();

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

        for(int i = 0; i < n; i++) {
            System.out.println("Enter details for book " + (i+1) + ":");
            System.out.println("Name: ");
            String name = scan.nextLine();

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

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

            System.out.println("Number of pages: ");
            int numPages = scan.nextInt();

            Book book = new Book(name, author, price, numPages);
            books.add(book);
        }

        System.out.println("\n Book Details: ");
        for(Book book : books) {
            System.out.println(book);
        }
        scan.close();
    }
}
```

Output:

```
Enter the number of books: 2
Enter details for book 1:
Name: Harry potter
Author: harry
Price: 78
Number of Pages: 345
Enter details for book 2:
Name: Geronimo
Author: Stilton
Price: 34
Number of Pages: 120

Details of the books:

Book name: Harry Potter
Author: harry
Price: $ 78.0
Number of Pages: 345

Book Name: Geronimo
Author: Stilton
Price: $ 34.0
Number of Pages: 120
```



OUTPUT 3

Program 4

Area of Shapes using Abstract class

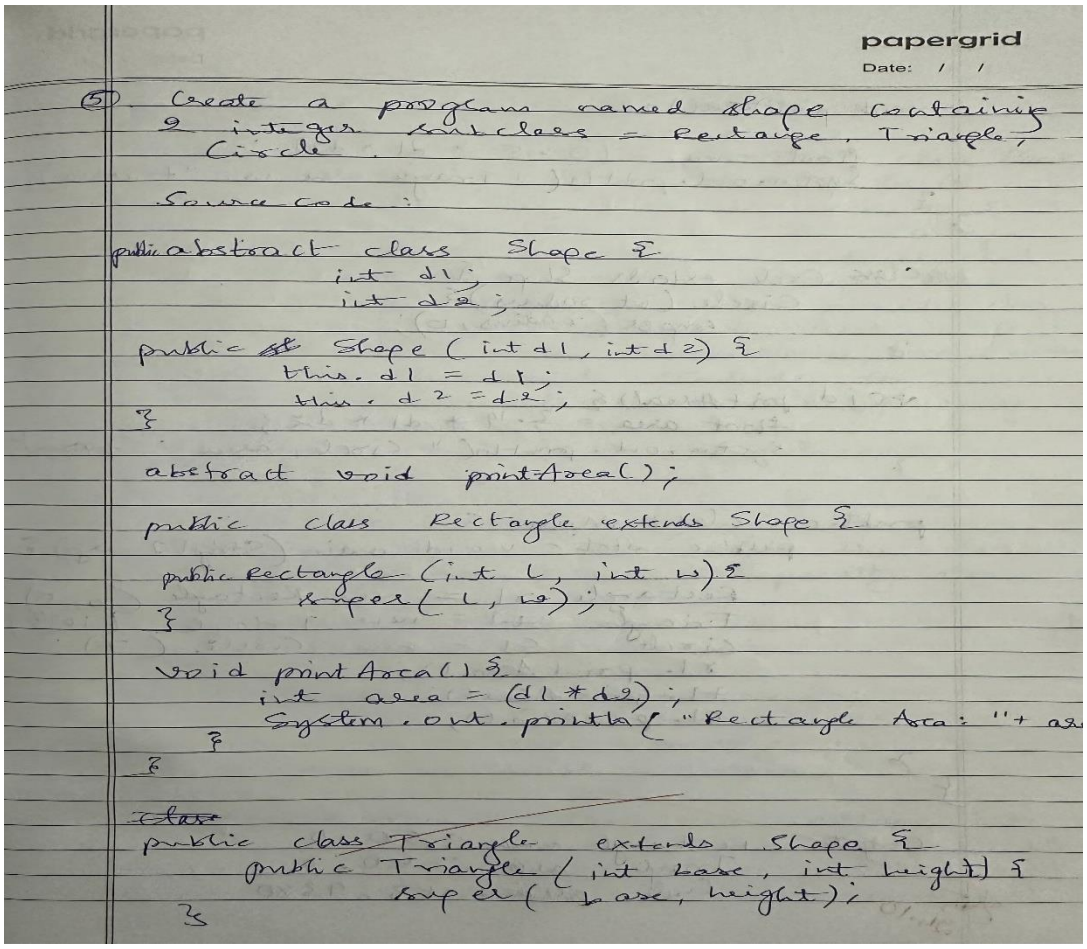
Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea()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

Source code:

```
public abstract class Shape{
    double a;
    double b;
    public Shape(double a, double b){
        this.a = a;
        this.b = b;
    }
    public abstract void printArea();
}
public class Rectangle extends Shape {
    public Rectangle(double a, double b){
```

```
        super(a,b);
    }
    @Override
    public void printArea(){
        double area = a * b;
        System.out.println("The area of the rectangle is: "+area);
    }
}
public class Triangle extends Shape {
    public Triangle(double a, double b){
        super(a,b);
    }
    @Override
    public void printArea(){
        double area = (0.5 * a * b);
        System.out.println("The area of the triangle is: "+area);
    }
}
public class Circle extends Shape {
    public Circle(double a, double b){
        super(a,b);
    }
    @Override
    public void printArea(){
        double area = Math.PI * a * a;
        System.out.println("The area of the Circle is: "+area);
    }
}
public class Main {
    public static void main(String[] args){
        Rectangle rectangle = new Rectangle(10, 20);
        Triangle trinagle = new Triangle(10, 20);
        Circle circle = new Circle(10, 20);
        rectangle.printArea();
        trinagle.printArea();
        circle.printArea();
```

```
  }
}
```

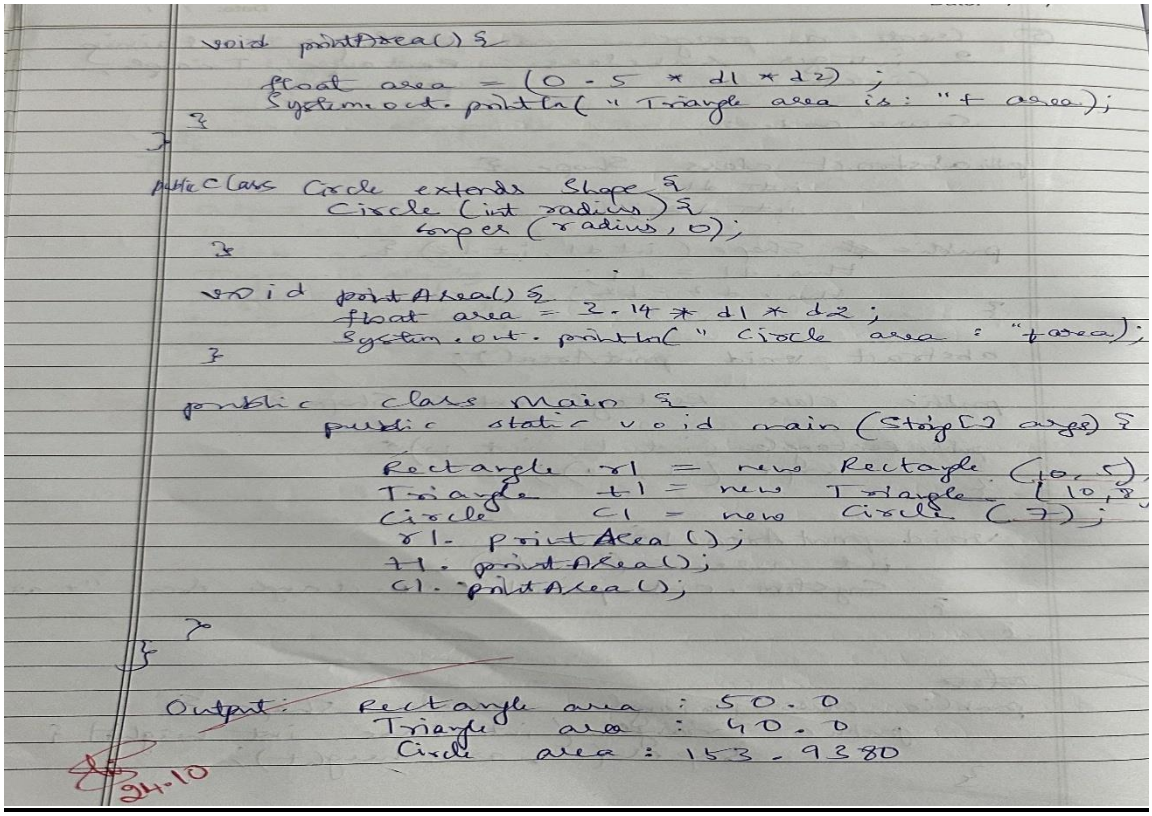


papergrid
Date: / /

(5) Create a program named shape containing 2 integer and subclass = Rectangle, Triangle, Circle.

Source code :

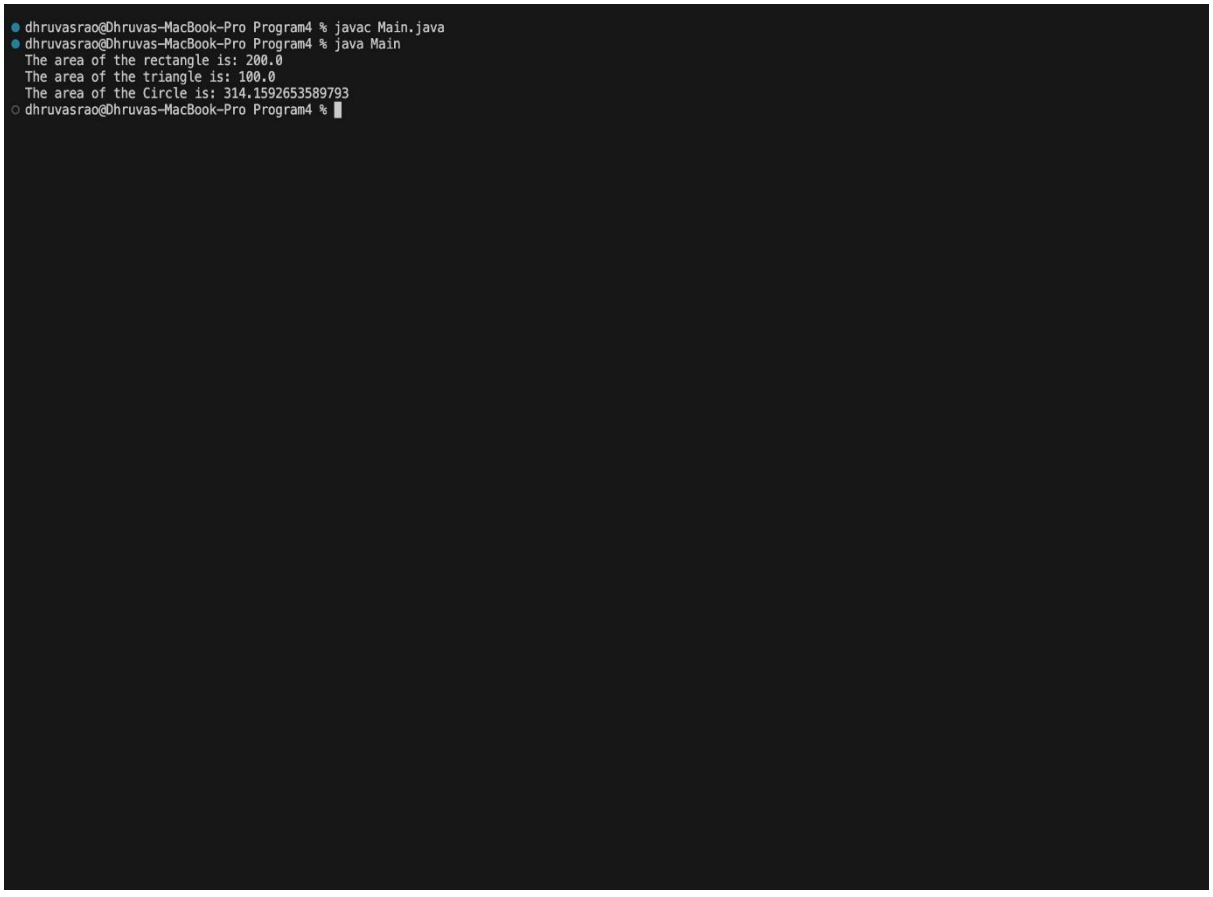
```
public abstract class Shape {
        int d1;
        int d2;

public Shape (int d1, int d2) {
        this.d1 = d1;
        this.d2 = d2;
}

abstract void printArea();

public class Rectangle extends Shape {

public Rectangle (int l, int w) {
        super(l, w);
}

void printArea() {
        int area = (d1 * d2);
        System.out.println("Rectangle Area: "+ ar
}
}

public class Triangle extends Shape {
        public Triangle (int base, int height) {
                super(base, height);
}
```



```
void printArea() {
    float area = (0.5 * d1 * d2);
    System.out.println("Triangle area is: " + area);
}
}

public class Circle extends Shape {
    Circle(int radius) {
        super(radius, 0);
    }

    void printArea() {
        float area = 3.14 * d1 * d2;
        System.out.println("Circle area: " + area);
    }

public class Main {
    public static void main(String[] args) {
        Rectangle r1 = new Rectangle(10, 5);
        Triangle t1 = new Triangle(10, 8);
        Circle c1 = new Circle(7);
        r1.printArea();
        t1.printArea();
        c1.printArea();
    }
}
```

Output:
Rectangle area : 50.0
Triangle area : 40.0
Circle area : 153.9380

24.10



```
dhruvasrao@Dhruvas-MacBook-Pro Program4 % javac Main.java
dhruvasrao@Dhruvas-MacBook-Pro Program4 % java Main
The area of the rectangle is: 200.0
The area of the triangle is: 100.0
The area of the Circle is: 314.1592653589793
dhruvasrao@Dhruvas-MacBook-Pro Program4 %
```

OUTPUT 4

Program 5

Bank Account

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.

Source code:

```
import java.util.Scanner;

public class Bank {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        SavAcct savings = new SavAcct("Alice", "S12345", 5000);

        CurAcct current = new CurAcct("Bob", "C12345", 2000);

        System.out.println("Savings Account:");

        savings.displayBalance();

        savings.deposit(1000);

        savings.displayBalance();

        savings.computeAndDepositInterest();

        savings.displayBalance();

        savings.withdraw(2000);

        savings.displayBalance();

        System.out.println("\nCurrent Account:");

        current.displayBalance();

        current.deposit(500);

        current.displayBalance();

        current.withdraw(2500);

        current.displayBalance();

        current.withdraw(300);

        current.displayBalance();

        scanner.close();
```

```
    }
}
class SavAcct extends Account {
    private static final double INTEREST_RATE = 0.04;

    public SavAcct(String customerName, String accountNumber, double initialBalance) {
        super(customerName, accountNumber, "Savings", initialBalance);
    }
    public void computeAndDepositInterest() {
        double interest = balance * INTEREST_RATE;
        balance += interest;
        System.out.println("Interest deposited: " + interest);
    }
    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount);
        } else {
            System.out.println("Invalid withdrawal amount or insufficient balance.");
        }
    }
}
class CurAcct extends Account {
    private static final double MIN_BALANCE = 1000.0;
    private static final double PENALTY = 50.0;

    public CurAcct(String customerName, String accountNumber, double initialBalance) {
        super(customerName, accountNumber, "Current", initialBalance);
    }
    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount);
            checkMinimumBalance();
        } else {
            System.out.println("Invalid withdrawal amount or insufficient balance.");
```

```
        }
    }
    private void checkMinimumBalance() {
        if (balance < MIN_BALANCE) {
            balance -= PENALTY;
            System.out.println("Balance fell below minimum. Penalty imposed: " + PENALTY);
        }
    }
}
class Account {
    protected String customerName;
    protected String accountNumber;
    protected String accountType;
    protected double balance;
    public Account(String customerName, String accountNumber, String accountType, double initialBalance) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = initialBalance;
    }
    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);
    }
}
```

papergrid
Date: / /

Lab Program – 6

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 account provides the compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Create a class Account that stores CustomerName, account number and type of account. 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 minimum balance and penalty also.

Source code:

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

    public Account (String customerName, int
        accountNumber, String accountType, double
        balance) {
            this.customerName = customerName
```

Date: / /

```
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = balance;

    public void deposit (double amount) {
        balance += amount;
        System.out.println ("Amount
            deposited: "+amount);
        System.out.println ("Updated balance: "+
            balance);
    }

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

class SavAcct extends Account {
    private final double interestRate = 0.04;

    public SavAcct (String customerName, int
        accountNumber, double balance) {
        super (customerName, accountNumber,
            "Savings", balance);
    }

    public void computeAndDepositInterest () {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println ("Interest computed
            and added: "+interest);
```

papergrid
Date: / /

```
        System.out.println ("Updated balance after
            interest: "+balance);
    }

    public void withdraw (double amount) {
        if (amount <= balance) {
            balance -= amount;
            System.out.println ("Amount withdrawn:
                "+amount);
            System.out.println ("Updated balance: "+
                balance);
        } else {
            System.out.println ("Insufficient balance");
        }
    }
}

class CurAcct (String customerName, int accountNumber, "Current", double balance) {
    public CurAcct (String customerName, int accNo, double balance) {
        super (customerName, accountNumber, "Current",
            balance);
    }

    private void checkMinimumBalance () {
        if (balance < minimumBalance) {
            balance -= serviceCharge;
            System.out.println ("Service charge
                imposed is: "+serviceCharge);
        }
    }

    public void withdraw (double amount) {
```

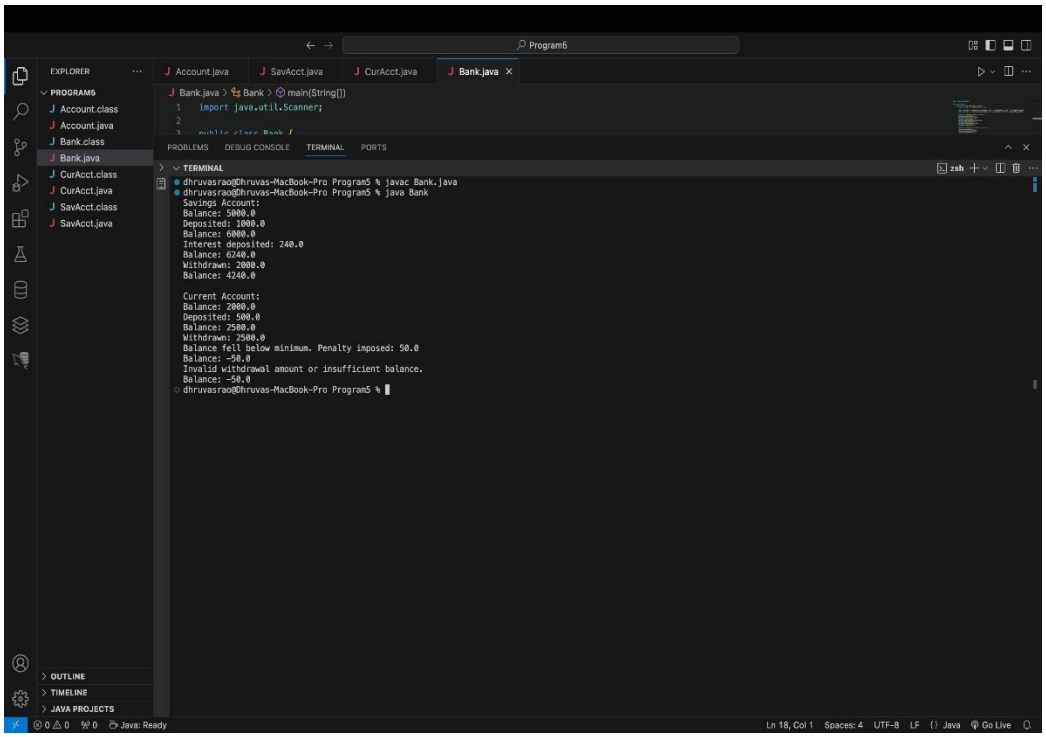
```
Date: / /

if (amount <= balance) {
    balance -= amount;
    System.out.println("Amount is: " +
        amount);
    checkMinimumBalance();
} else {
    System.out.println("Insufficient
        balance!");
}
}

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

        SavAcct savAcc = new SavAcct("A",
                123, 2000.0);
        savAcc.displayBalance();
        savAcc.deposit(500.0);

        savAcc.computeAndDepositInterest();
        savAcc.withdraw(300.0);

        CurAcct curAcc = new CurAcct("B",
                67890, 600.0);
        curAcc.displayBalance();
        curAcc.deposit(200.0);
        curAcc.withdraw(400.0);
        curAcc.withdraw(500.0);
    }
}
```

papergrid

Date: / /

Output:

```
Current balance: 2000.0
Amount deposited: 500.0
Updated balance: 2500.0
Amount withdrawn: 300.0
Updated balance: 2200.0
Service charge is: 50.0
Updated balance after service charge: 350.0
Insufficient balance for withdrawal!
```



OUTPUT 5

Program 6

CIE and SEE Packages Implementation

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.

Source code:

```
package CIE;
public class Internals {
    public int[] internalMarks = new int[5];
    public Internals(int[] marks) {
        if (marks.length == 5) {
            System.arraycopy(marks, 0, internalMarks, 0, 5);
        } else {
            throw new IllegalArgumentException("There must be exactly 5 internal marks.");
        }
    }
}
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;
    }
}
package SEE;
import CIE.Student;
```

```
public class External extends Student {
    public int[] seeMarks = new int[5];
    public External(String usn, String name, int sem, int[] marks) {
        super(usn, name, sem);
        if (marks.length == 5) {
            System.arraycopy(marks, 0, seeMarks, 0, 5);
        } else {
            throw new IllegalArgumentException("There must be exactly 5 SEE marks.");
        }
    }
}
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 the number of students: ");
        int n = scanner.nextInt();
        scanner.nextLine();
        External[] students = new External[n];
        Internals[] internalMarks = new Internals[n];
        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for student " + (i + 1));
            System.out.print("USN: ");
            String usn = scanner.nextLine();
            System.out.print("Name: ");
            String name = scanner.nextLine();
            System.out.print("Semester: ");
            int sem = scanner.nextInt();
            System.out.print("Enter 5 internal marks: ");
            int[] internal = new int[5];
            for (int j = 0; j < 5; j++) {
                internal[j] = scanner.nextInt();
            }
            internalMarks[i] = new Internals(internal);
```

```
            System.out.print("Enter 5 SEE marks: ");
            int[] external = new int[5];
            for (int j = 0; j < 5; j++) {
                external[j] = scanner.nextInt();
            }
            scanner.nextLine();
            students[i] = new External(usn, name, sem, external);
            System.out.println();
        }
        System.out.println("Final Marks:");
        for (int i = 0; i < n; i++) {
            System.out.println("Student: " + students[i].name);
            for (int j = 0; j < 5; j++) {
                int finalMark = internalMarks[i].internalMarks[j] + students[i].seeMarks[j];
                System.out.println("Course " + (j + 1) + " Final Mark: " + finalMark);
            }
            System.out.println();
        }
        scanner.close();
    }
}
```

Lab Program 7

Create a package CIE which has two classes - Student and Internals. The class Personal has members like usn, name, sem. The class 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 students. Import the two packages in a file that declares the final marks of n students in all five courses.

```
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 display() {
        System.out.println("USN: " + usn + "\nName: " + name + "\nSemester: " + sem);
    }
}
package CIE;
```

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

    public Internals(int[] marks) {
        Arrays.copyOf(this.marks, this.marks.length);
    }
}
import CIE.*;
import SEE.*;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the 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("Enter name: \nUSN: \nSemester: ");
        }

        for(int i = 0; i<n; i++) {
            System.out.println("Enter USN for student: " + (i+1) + ": ");
            String usn = scanner.nextLine();
            System.out.print("Enter name: " + (i+1) + ": ");
            String name = scanner.nextLine();
            System.out.print("Enter semester: " + (i+1) + ": ");
            int sem = scanner.nextInt();

            int[] internalMarks = new int[5];
```

```
            System.out.println("Enter external marks: " + (i+1) + ": ");
            for(int j=0; j<5; j++) {
                externalMarks[j] = scanner.nextInt();
            }
            External external = new External(usn, name, sem, externalMarks);
            int[] finalMarks = external.calculateFinalMarks(internals);
            System.out.println("\nFinal marks " + (i+1) + " (" + name + "): ");
            for(int j = 0; j<5; j++) {
                System.out.println("Course " + (j+1) + ": " + finalMarks[j]);
            }
        }
        scanner.close();
    }
}

package SEE;
import CIE.Student;
import CIE.Internals;

public class External extends Student {
    private int[] externalMarks = new int[5];

    public External(String usn, String name, int sem, int[] externalMarks) {
        super(usn, name, sem);
        if(externalMarks.length == 5) {
            this.externalMarks = externalMarks;
        } else {
            System.out.println("Invalid");
        }
    }
```

```
    public int[] getExternalMarks() {
        return externalMarks;
    }

    public int[] calculateFinalMarks(Internals internals) {
        int[] finalMarks = new int[5];
        for(int i=0; i<5; i++) {
            finalMarks[i] = (internals.getInternalMarks()[i] + this.externalMarks[i]) / 2;
        }
        return finalMarks;
    }
}
```

Output:

```
Enter the number of students: 1
Enter name: John
Enter usn: 2BM4CS081
Enter semester: 2
Enter internal marks: 24
                      35
                      40
                      40
                      21
Enter external marks: 98
                      97
                      55
                      37
                      94
Final marks of John is: 81
                        87
                        80
                        65
```

```
dhruvasrao@Dhruvas-MacBook-Pro CIE % javac Main.java
dhruvasrao@Dhruvas-MacBook-Pro CIE % java Main
Enter the number of students: 1
Enter details for student 1
USN: 1BM23CS988
Name: John
Semester: 3
Enter 5 internal marks: 35
55
89
12
67
Enter 5 SEE marks: 99
67
55
88
90

Final Marks:
Student: John
Course 1 Final Mark: 134
Course 2 Final Mark: 122
Course 3 Final Mark: 144
Course 4 Final Mark: 100
Course 5 Final Mark: 157

dhruvasrao@Dhruvas-MacBook-Pro CIE %
```

OUTPUT 6

Program 7

Exception Handling

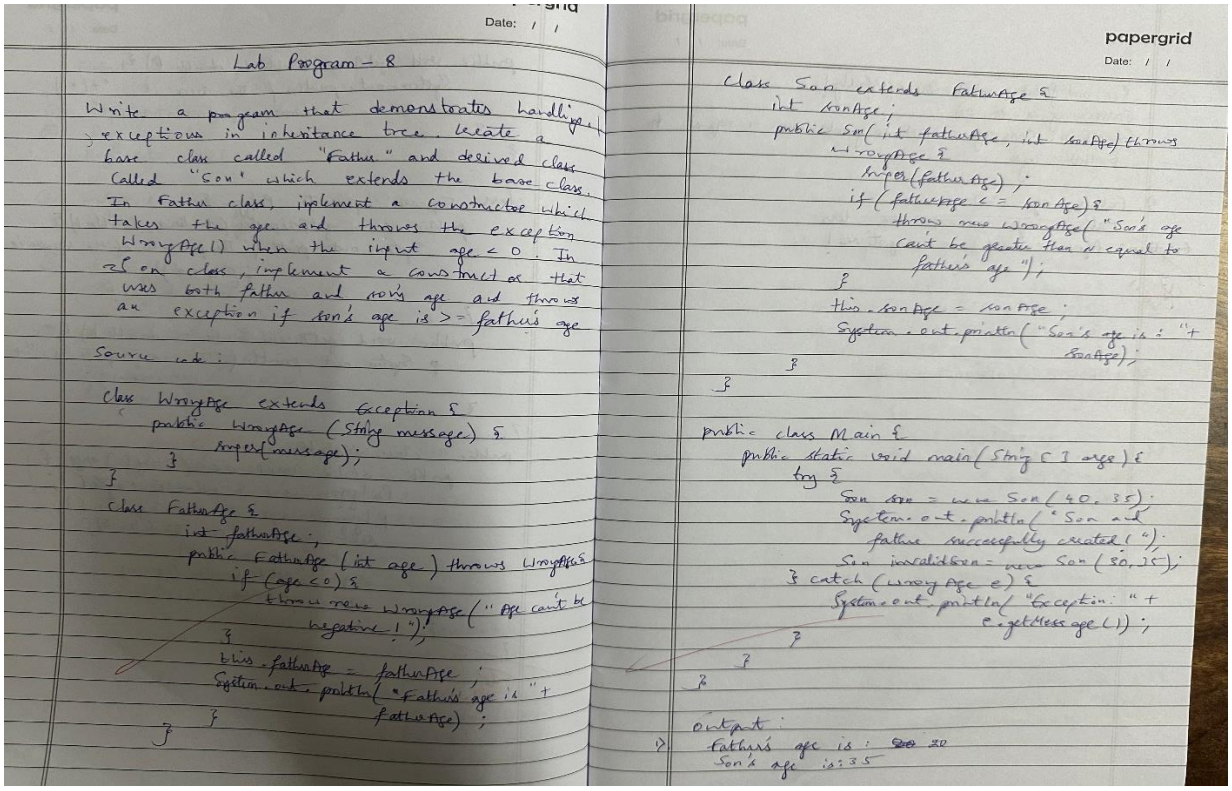
Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age=father’s age.

Source code:

```
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("Age cannot be negative for Father.");
        }
        this.age = age;
```

```
    }
    public int getAge(){
        return age;
    }
}
class Son extends Father {
     public int sonAge;
    public Son(int fatherAge, int sonAge) throws WrongAgeException {
        super(fatherAge);
        if (sonAge < 0) {
            throw new WrongAgeException("Age cannot be negative for Son.");
        }
        if (sonAge >= fatherAge) {
            throw new WrongAgeException("Son's age cannot be greater than or equal to Father's age.");
        }
        this.sonAge = sonAge;
    }
}
public class Main{
    public static void main(String[] args) {
        try {
            Father father = new Father(45);
            System.out.println("Father's age is valid.");

            Son son = new Son(45, 20);
            System.out.println("Son's age is valid.");
        } catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }
        try {
            Father invalidFather = new Father(-5);
        } catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }
        try {
            Son invalidSon = new Son(40, 45);
```

```
        } catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```



Lab Program – 8

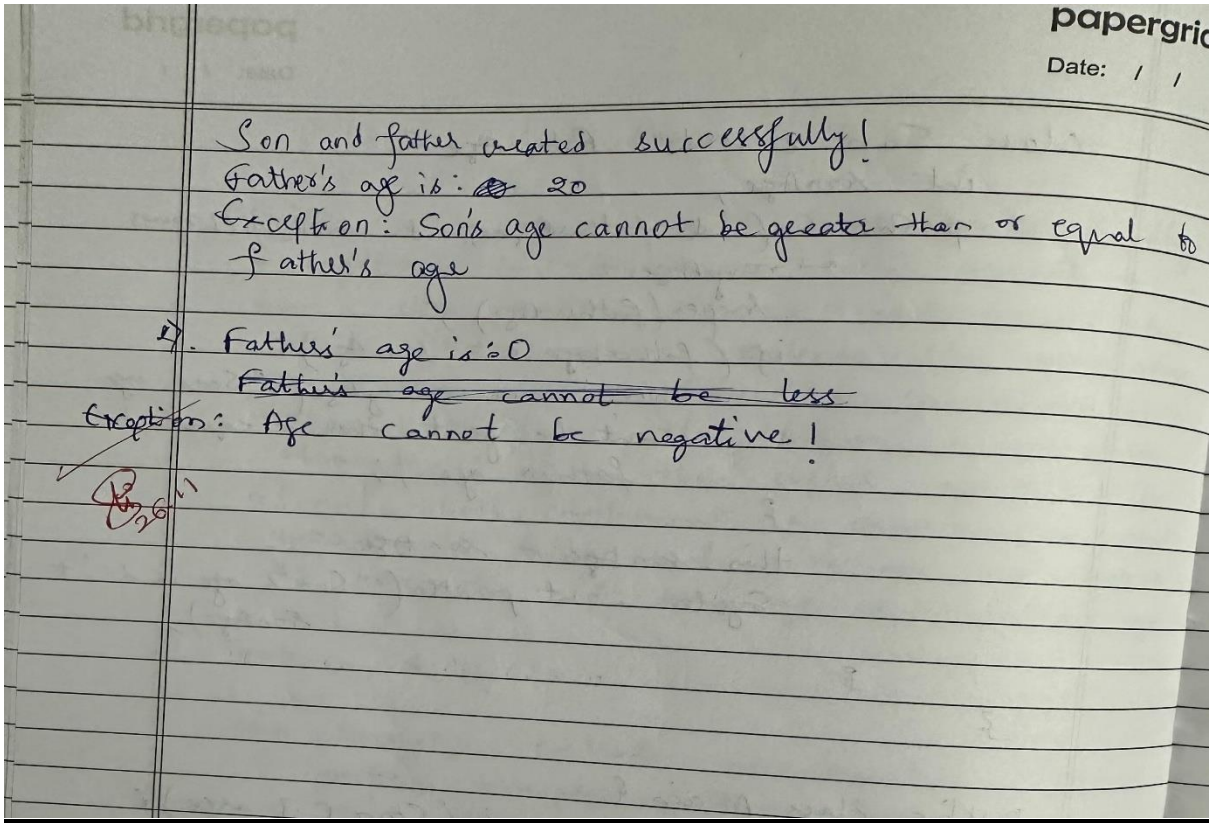
Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and derived class called "Son" which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAgeException when the input age < 0.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

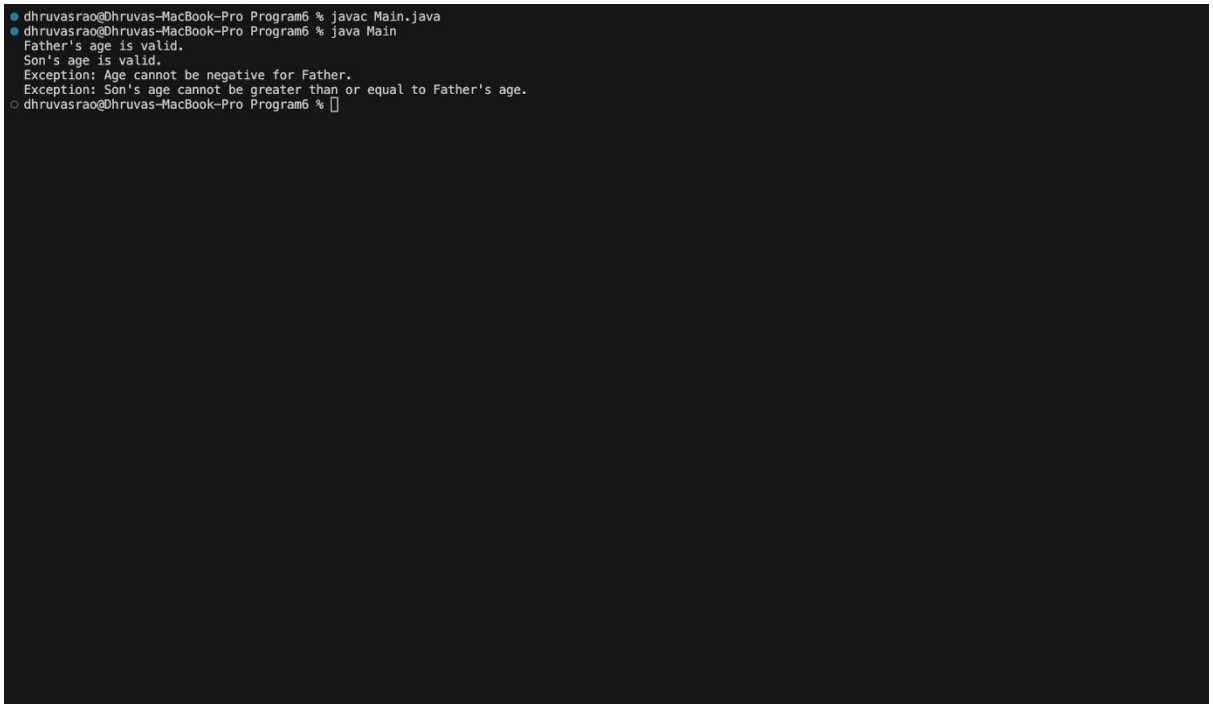
Source code:

```
class WrongAge extends Exception {
    public WrongAge(String message) {
        super(message);
    }
}
class FatherAge {
    int fatherAge;
    public FatherAge(int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge("Age can't be negative!");
        }
        this.fatherAge = fatherAge;
        System.out.println("Father's age is: " + fatherAge);
    }
}
class Son extends FatherAge {
    int sonAge;
    public Son(int fatherAge, int sonAge) throws WrongAge {
        super(fatherAge);
        if (fatherAge <= sonAge) {
            throw new WrongAge("Son's age can't be greater than or equal to father's age");
        }
        this.sonAge = sonAge;
        System.out.println("Son's age is: " + sonAge);
    }
}
public class Main {
    public static void main(String[] args) {
        try {
            Son son = new Son(40, 35);
            System.out.println("Son and father successfully created");
            Son invalidSon = new Son(30, 35);
        } catch (WrongAge e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```

output:

```
Father's age is: 40
Son's age is: 35
```


papergri
Date: / /
Son and father created successfully!
Father's age is: 20
Exception: Son's age cannot be greater than or equal to Father's age
1). Father's age is: 0
Father's age cannot be less
Exception: Age cannot be negative!


dhruvasrao@Dhruvas-MacBook-Pro Program6 % javac Main.java
dhruvasrao@Dhruvas-MacBook-Pro Program6 % java Main
Father's age is valid.
Son's age is valid.
Exception: Age cannot be negative for Father.
Exception: Son's age cannot be greater than or equal to Father's age.
dhruvasrao@Dhruvas-MacBook-Pro Program6 %

OUTPUT 7

Program 8

Thread Program Implementation

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

Source code:

```
class DisplayMessage extends Thread {
    private String message;
    private int interval;

    public DisplayMessage(String message, int interval) {
        this.message = message;
        this.interval = interval;
    }
    @Override
    public void run() {
        try {
            while (true) {
                System.out.println(message);
                Thread.sleep(interval);
            }
        } catch (InterruptedException e) {
            System.out.println("Thread interrupted: " + message);
        }
    }
}
public class Main {
    public static void main(String[] args) {
        Thread thread1 = new DisplayMessage("BMS College of Engineering", 10000);
        Thread thread2 = new DisplayMessage("CSE", 2000);
        thread1.start();
        thread2.start();
    }
}
```

papergrid

Date: / /

Lab Program – 9

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.

Source code:

```
public class BmsCseThread {
    public static void main(String[] args) {
        Thread t1 = new Thread(() -> {
            while (true) {
                System.out.println("BMS College Of
                                    Engineering!");
                try {
                    Thread.sleep(10000);
                } catch (InterruptedException e) {
                    System.out.println("Thread 1
                                        interrupted!");
                }
            }
        });
        Thread t2 = new Thread(() -> {
            while (true) {
                System.out.println("CSE!");
                try {
                    Thread.sleep(2000);
                } catch (InterruptedException e) {
                    System.out.println("Thread 2
                                        interrupted");
                }
            }
        });
```

Date

```
        t1.start();
        t2.start();
    }
}
```

Output

```
BMS College of Engineering!
CSE!
CSE!
CSE!
CSE!
CSE!
BMS College of Engineering!
:
:
(It repeats)
```

OUTPUT 8

Program 9

Interface Implementation

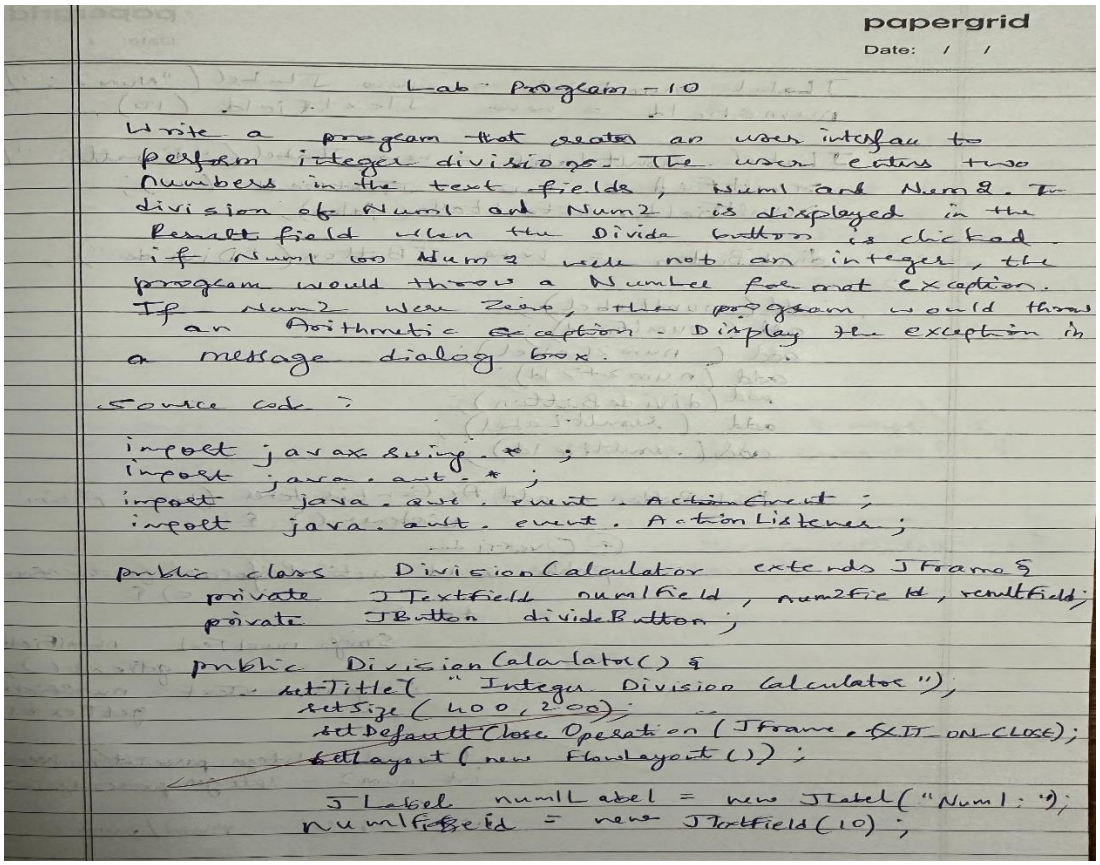
Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException.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

Source code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
public class IntegerDivisionUI {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Integer Division");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 200);
        JLabel label1 = new JLabel("Num1:");
        JTextField num1Field = new JTextField(10);
        JLabel label2 = new JLabel("Num2:");
```

```
        JTextField num2Field = new JTextField(10);
        JButton divideButton = new JButton("Divide");
        JTextField resultField = new JTextField(15);
        resultField.setEditable(false);
        divideButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    int num1 = Integer.parseInt(num1Field.getText());
                    int num2 = Integer.parseInt(num2Field.getText());
                    if (num2 == 0) {
                        throw new ArithmeticException("Division by zero");
                    }
                    int result = num1 / num2;
                    resultField.setText(String.valueOf(result));
                } catch (NumberFormatException ex) {
                    JOptionPane.showMessageDialog(frame, "Please enter valid integers.", "Input Error",
JOptionPane.ERROR_MESSAGE);
                } catch (ArithmeticException ex) {
                    JOptionPane.showMessageDialog(frame, "Cannot divide by zero.", "Arithmetic Error",
JOptionPane.ERROR_MESSAGE);
                }
            }
        });
        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(4, 2, 5, 5));
        panel.add(label1);
        panel.add(num1Field);
        panel.add(label2);
        panel.add(num2Field);
        panel.add(new JLabel("Result:"));
        panel.add(resultField);
        panel.add(new JLabel());
        panel.add(divideButton);
        frame.add(panel);
        frame.setVisible(true);
    }
```

}



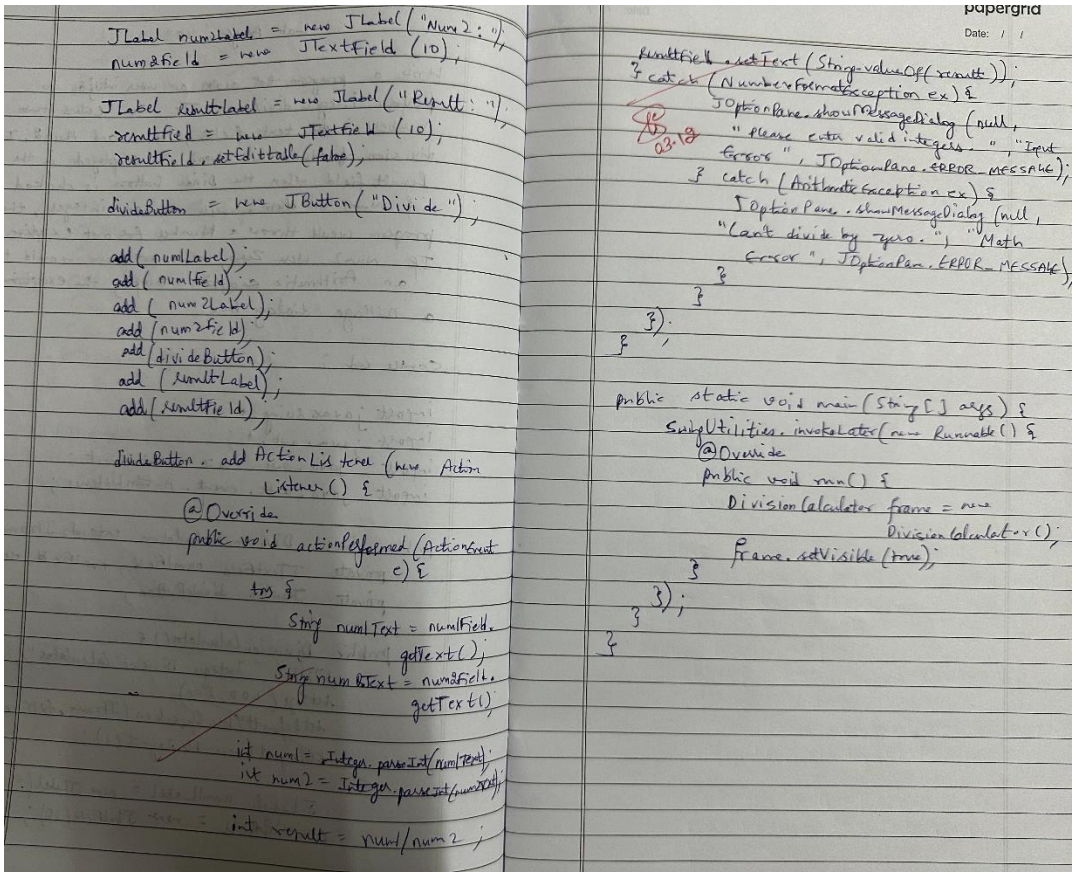
papergrid
Date: / /

Lab Program - 10

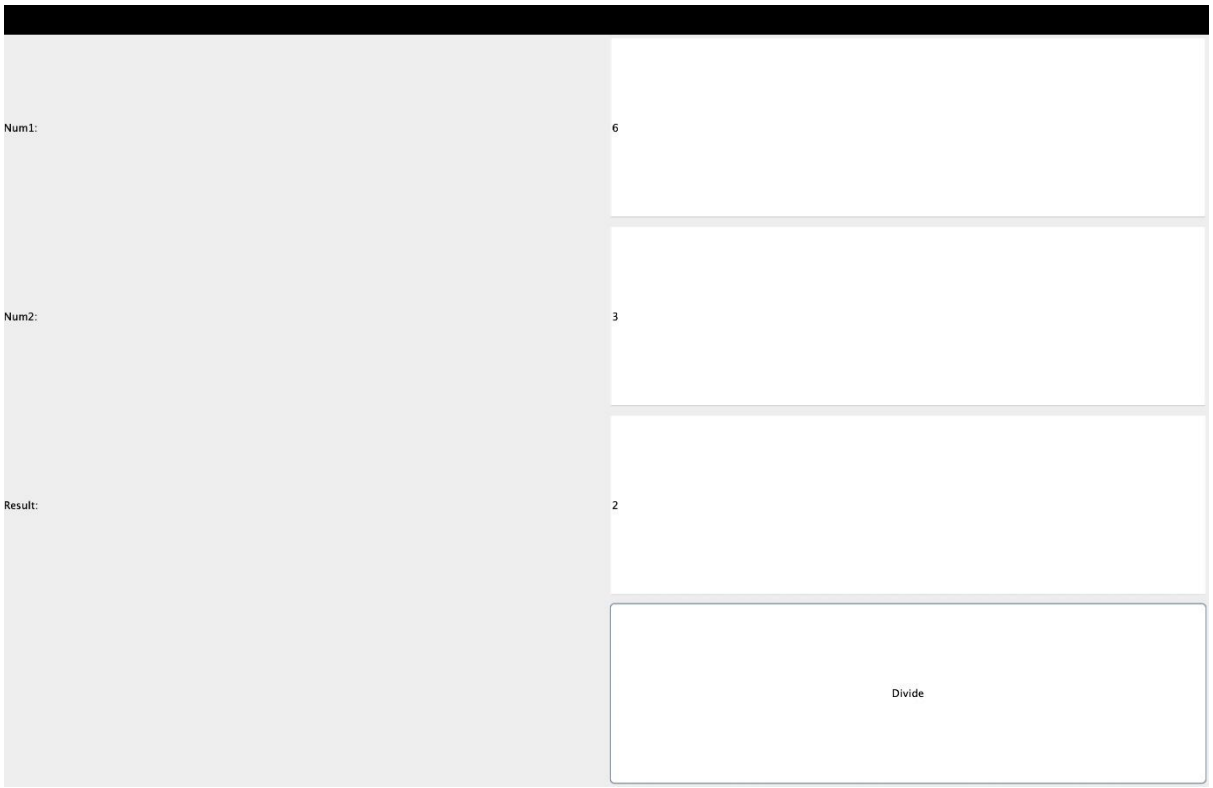
Write a program that creates an user interface to perform integer division. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException.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

Source code :

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class DivisionCalculator extends JFrame {
    private JTextField num1Field, num2Field, resultField;
    private JButton divideButton;

    public DivisionCalculator() {
        setTitle("Integer Division Calculator");
        setSize(400, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new FlowLayout());

        JLabel num1Label = new JLabel("Num1:");
        num1Field = new JTextField(10);
```



```
        JLabel num2Label = new JLabel("Num2:");
        num2Field = new JTextField(10);

        JLabel resultLabel = new JLabel("Result:");
        resultField = new JTextField(10);
        resultField.setEditable(false);

        divideButton = new JButton("Divide");

        add(num1Label);
        add(num1Field);
        add(num2Label);
        add(num2Field);
        add(divideButton);
        add(resultLabel);
        add(resultField);

        divideButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    String num1Text = num1Field.getText();
                    String num2Text = num2Field.getText();

                    int num1 = Integer.parseInt(num1Text);
                    int num2 = Integer.parseInt(num2Text);

                    int result = num1 / num2;
                    resultField.setText(String.valueOf(result));
                } catch (NumberFormatException ex) {
                    JOptionPane.showMessageDialog(null,
                        "Please enter valid integers.", "Input
                        Error", JOptionPane.ERROR_MESSAGE);
                } catch (ArithmeticException ex) {
                    JOptionPane.showMessageDialog(null,
                        "Can't divide by zero.", "Math
                        Error", JOptionPane.ERROR_MESSAGE);
                }
            }
        });
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                DivisionCalculator frame = new
                    DivisionCalculator();
                frame.setVisible(true);
            }
        });
    }
}
```



OUTPUT 9

Program 10

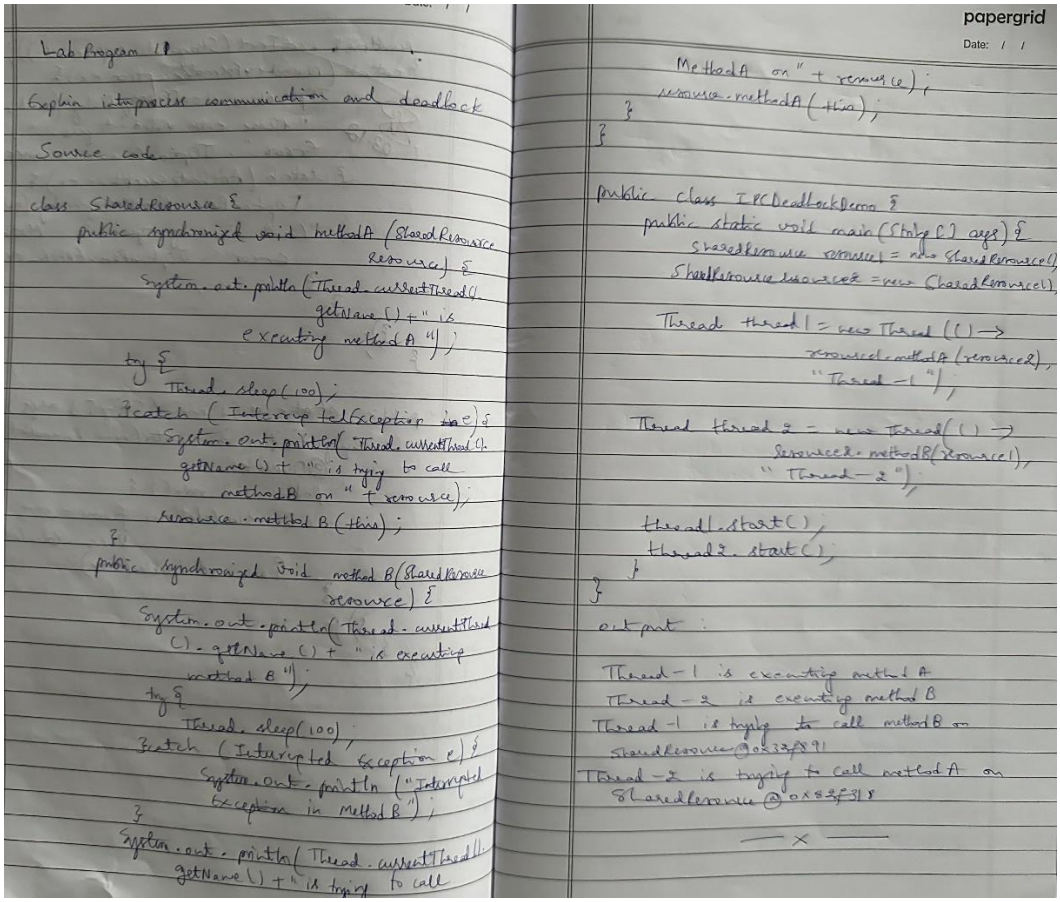
Inter Process Communication and Deadlock

Demonstrate Inter process Communication and deadlock

Source code:

```
class SharedResource {
    public synchronized void methodA(SharedResource resource) {
        System.out.println(Thread.currentThread().getName() + " is executing methodA");
        try {
            Thread.sleep(100);
        } catch (InterruptedException e) {
            System.out.println("InterruptedException in methodA");
        }
        System.out.println(Thread.currentThread().getName() + " is trying to call methodB on " + resource);
        resource.methodB(this);
    }
    public synchronized void methodB(SharedResource resource) {
```

```
        System.out.println(Thread.currentThread().getName() + " is executing methodB");
        try {
            Thread.sleep(100);
        } catch (InterruptedException e) {
            System.out.println("InterruptedException in methodB");
        }
        System.out.println(Thread.currentThread().getName() + " is trying to call methodA on " + resource);
        resource.methodA(this);
    }
}
public class IPCDeadlockDemo {
    public static void main(String[] args) {
        SharedResource resource1 = new SharedResource();
        SharedResource resource2 = new SharedResource();
        Thread thread1 = new Thread(() -> resource1.methodA(resource2), "Thread-1");
        Thread thread2 = new Thread(() -> resource2.methodB(resource1), "Thread-2");
        thread1.start();
        thread2.start();
    }
}
```



Lab Program 10

Explain interprocess communication and deadlock

Source code:

```
class SharedResource {
    public synchronized void methodA(SharedResource resource) {
        System.out.println(Thread.currentThread().getName() + " is executing methodA");
        try {
            Thread.sleep(100);
        } catch (InterruptedException e) {
            System.out.println(Thread.currentThread().getName() + " is trying to call methodB on " + resource);
            resource.methodB(this);
        }
    }
    public synchronized void methodB(SharedResource resource) {
        System.out.println(Thread.currentThread().getName() + " is executing methodB");
        try {
            Thread.sleep(100);
        } catch (InterruptedException e) {
            System.out.println("Interrupted exception in methodB");
        }
        System.out.println(Thread.currentThread().getName() + " is trying to call methodA on " + resource);
        resource.methodA(this);
    }
}

public class IPCDeadlockDemo {
    public static void main(String[] args) {
        SharedResource resource1 = new SharedResource();
        SharedResource resource2 = new SharedResource();

        Thread thread1 = new Thread(() -> resource1.methodA(resource2), "Thread-1");
        Thread thread2 = new Thread(() -> resource2.methodB(resource1), "Thread-2");

        thread1.start();
        thread2.start();
    }
}
```

output:

```
Thread-1 is executing methodA
Thread-2 is executing methodB
Thread-1 is trying to call methodB on SharedResource@0x32f891
Thread-2 is trying to call methodA on SharedResource@0x32f318
```

— x —

```
dhruvasrao@Dhruvas-MacBook-Pro Program9 % javac IPCDeadlockDemo.java
dhruvasrao@Dhruvas-MacBook-Pro Program9 % java IPCDeadlockDemo
Thread-1 is executing methodA
Thread-2 is executing methodB
Thread-2 is trying to call methodA on SharedResource@57a2eb1f
Thread-1 is trying to call methodB on SharedResource@3ed099d6
```

OUTPUT 10