VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Danish K (1BM23CS086)

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 **Danish Kodavanti (1BM23CS086),** 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.

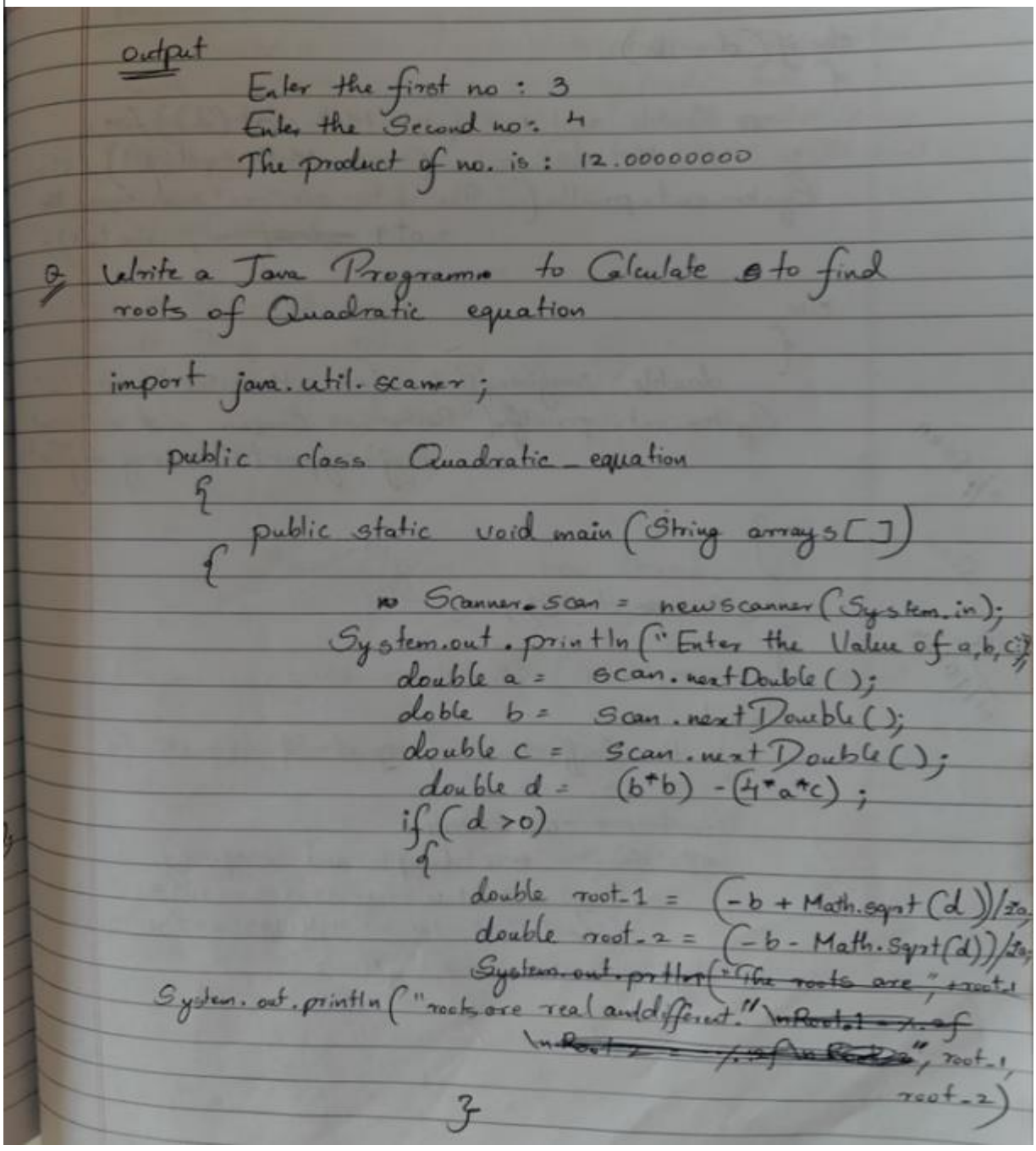
Lab faculty Incharge Name Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Github-link(https://github.com/Danishkodavanti/java_lab_sem3)

Program 1

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.



output

```
Enter the first no : 3
Enter the Second no: 4
The product of no. is : 12.00000000
```

Q. Write a Java Programme to Calculate to find roots of Quadratic equation

```
import java.util.scaner;

public class Quadratic_equation
{
    public static void main(String args[])
    {
        Scanner scan = new scanner(System.in);
        System.out.println("Enter the Value of a,b,c");
        double a = scan.nextDouble();
        doble b = scan.nextDouble();
        double c = scan.nextDouble();
        double d = (b*b) - (4*a*c);
        if(d>0)
        {
            double root_1 = (-b + Math.sqrt(d))/2a;
            double root_2 = (-b - Math.Sqrt(d))/2a;
            System.out.println("The roots are ",root_1
System.out.println("roots are real and different." \n Root 1 = ",root_1,
                                                    root_2)
    }
}
```

```
else if(d == 0)
{
    double root_1 = (-b + Math.sqrt(d)) / 2*a;
    double root_2 = (-b - Math.sqrt(d)) / 2*a;
System.out.println("The roots are real and equal
                    root: ", root_1);

else
{
    double imaginaryPart = (-b + Math.Sqrt(d)) / 2*
System.out.println("Roots are Complex and different
                    imaginary part:" + imaginaryPart

}
    Scanner.close();
}
```

o/p seen

seen

01/10/24

Output

Enter the Coefficients of quadratic equation (a, b, c):
1 2 3

~~Root 1 : -1.0 + 1.41421~~

The Roots are Comple and imaginary
Root 1 : -1.0 + 1.4142135623709 51 i
Root 2 : -1.0 - 1.4142135623730951 i

CODE:

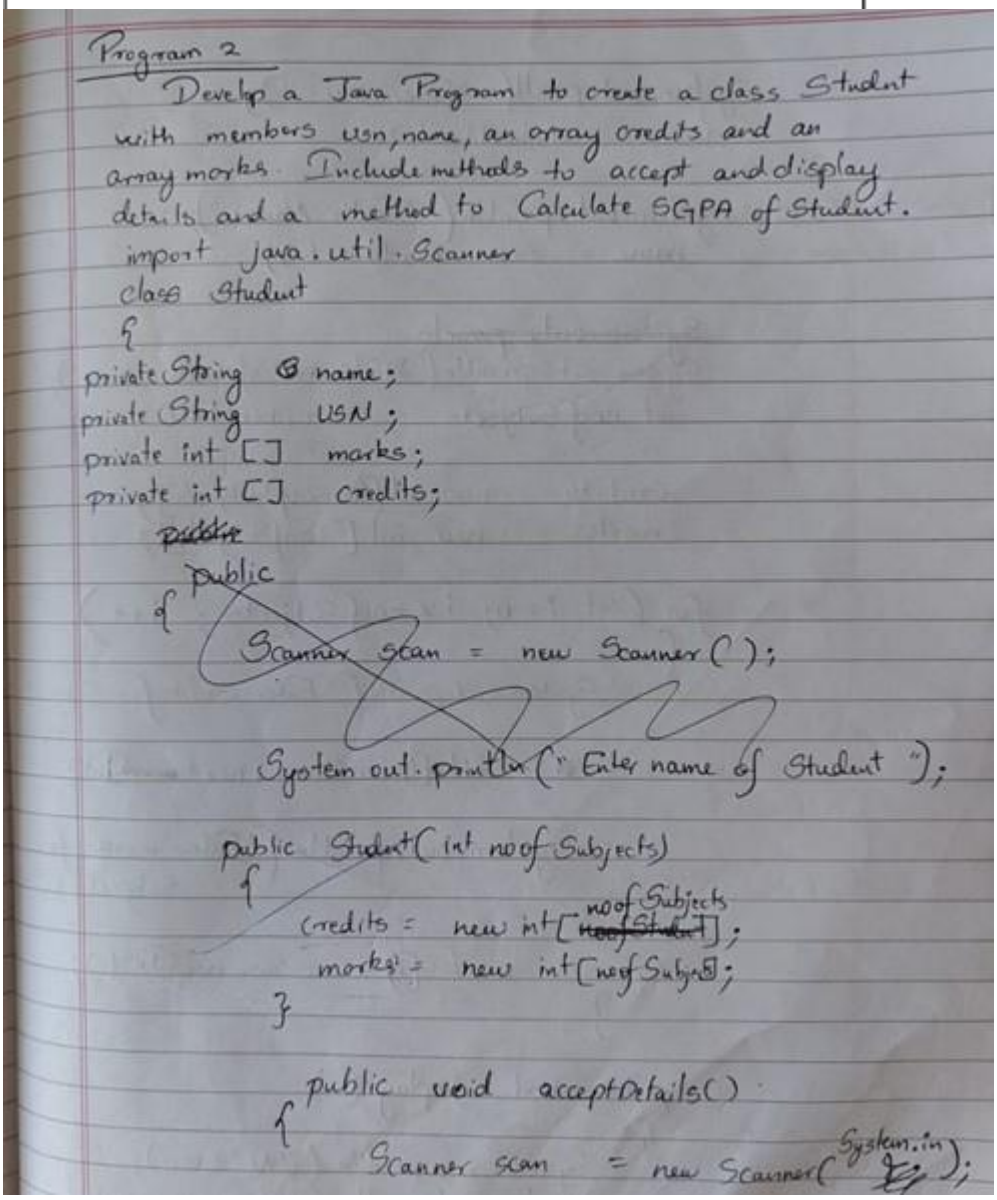
```
import java.util.Scanner;

public class QuadraticEquation {
    public static void main(String[] args) {
        // Scanner object for taking user input
        Scanner scanner = new Scanner(System.in);

        // Input values for a, b, and c
        System.out.print("Enter the value of a: ");
        double a = scanner.nextDouble();

        System.out.print("Enter the value of b: ");
        double b = scanner.nextDouble();

        System.out.print("Enter the value of c: ");
        double c = scanner.nextDouble();

        // Calculate the discriminant
        double discriminant = b * b - 4 * a * c;

        // Check if discriminant is positive, negative, or zero
        if (discriminant > 0) {
            // Two real and distinct roots
            double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
            double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);
            System.out.println("The roots are real and distinct.");
            System.out.println("Root 1: " + root1);
            System.out.println("Root 2: " + root2);
        } else if (discriminant == 0) {
            // One real root (both roots are the same)
            double root = -b / (2 * a);
            System.out.println("The roots are real and the same.");
            System.out.println("Root: " + root);
        } else {
            // Complex roots
            double realPart = -b / (2 * a);
            double imaginaryPart = Math.sqrt(-discriminant) / (2 * a);
            System.out.println("The roots are complex and distinct.");
            System.out.println("Root 1: " + realPart + " + " + imaginaryPart + "i");
```

```
            System.out.println("Root 2: " + realPart + " - " + imaginaryPart + "i");
        }

        System.out.println("Name: Danish k \n USN: 1BM23CS086");
        scanner.close();
    }
}
```

```
E:\danish_with_java\java_lab\quadratic_equations>java QuadraticEquation
Enter the value of a: 2
Enter the value of b: 3
Enter the value of c: 4
The roots are complex and distinct.
Root 1: -0.75 + 1.1989578808281798i
Root 2: -0.75 - 1.1989578808281798i
Name: Danish k
 USN: 1BM23CS086
```

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.



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 Student.

```
import java.util.Scanner
class Student
{
private String name;
private String USN;
private int [] marks;
private int [] credits;
public
{
    Scanner scan = new Scanner(System.in);
    System.out.println("Enter name of Student");
public Student(int noofSubjects)
{
    credits = new int[noofSubjects];
    marks = new int[noofSubjects];
}
public void acceptDetails()
{
    Scanner scan = new Scanner(System.in);
```

```
System.out.println("Enter USN: ");
usn = scan.nextLine();

System.out.println("Enter Name: ");
name = scan.nextLine();

System.out.println("Enter no. of Subjects: ");
int noofSubjects = scan.nextInt();

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

for(int i=0; i<noofSubjects; i++)
{
    System.out.println("Enter credits for Subject" +(i+1));
    credits[i]= scan.nextInt();

    System.out.println("Enter marks for Subject" +(i+1));
    marks[i] = scan.nextInt();
}

public void display()
{
    System.out.println("USN" + usn);
    System.out.println("Name" + name);
    ("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(    )
{
    ~~if(marks[i] >= 90)~~
    double totalPoints = 0;
    int totalCredits = 0;
    double SGPA = 0;
    for (int i = 0; i < credits.length; i++)
    {
        double gradePoint = 0;

        if(marks[i] >=90)
        {
            gradePoint = 10;
        }
        else if (marks[i] >=80)
        {
            gradePoint =9;
        }
        else if (marks[i] >=70)
        {     gradePoint = 8;}
        else if (marks[i] >=60)
        { gradePoint = 7; }
```

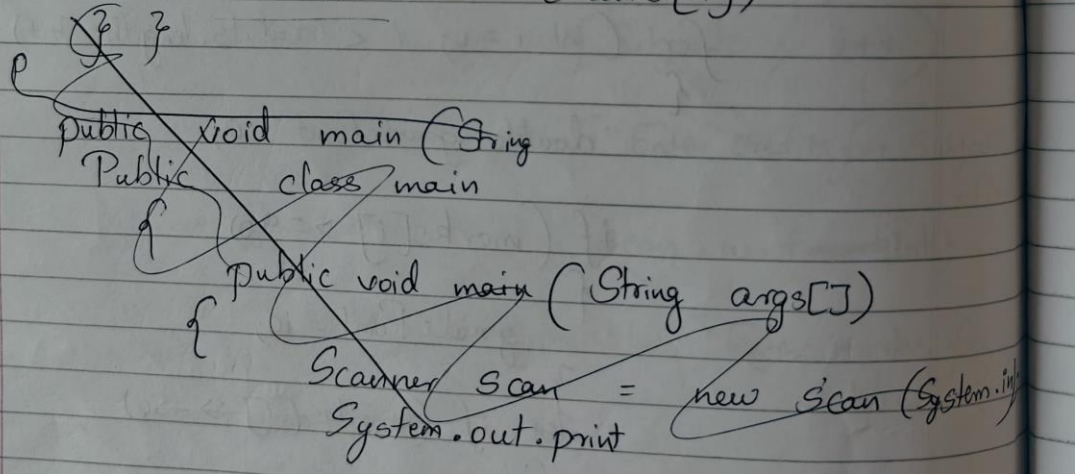
else if (marks[i] >= 50)
{ gradePoint = 6; }

else if (marks[i] >= 40
{ gradePoint = 5; }

else if (marks[i] < 40
{ gradePoint = 0; }

else
}

totalPoints += gradePoint * credits[i];
totalCredits += credits[i];



~~System.out.println~~

Soon proceed

SGPA = totalPoints / totalCredits;

System.out.println(" SGPA = "+(SGPA);

}

classmate
Date
Page

O/P

Entering details for Student 1:
Enter USN: 1BM23CS086
Enter Name: Danish
Credits for Sub 1 : 4
Marks for Sub 1 : 100
Credits for Sub 2 : 4
Marks ~~Credits~~ for Sub 2 : 98
Credits for Sub 3 : 3
Marks for Sub 3 : 78
Credits for Sub 4 : 3
Marks for Sub 4 : 100
Credits for Sub 5 : 3
Marks for Sub 6 : 98
Credits for Sub 7 : 2
Marks for Sub 7 : 78
Credits for Sub 8 : 2
Marks for Sub 8 : 67

Entering details for Student 2
Enter USN: 1BM23CS087
Enter Name: Darshan
Credits for Sub 1 : 4
Enter Marks for Sub 1 : 98
Credits for Sub 2 4
Marks for Sub 2 87
Credits for Sub 3 3
Marks for Sub 3 67
Credits for Sub 4 3

Credits for Sub5 : 2
Marks for Sub5 : 86
Credits for Sub6 : 2
Marks for Sub6 : 100
Credits for Sub7 : 1
Marks for Sub7 : 100
Credits for Sub8 : 1
Marks for Sub8 : 91

Details for Student 1
USN : 1BM23CS086
Name : Danish
Credits : 4 4 3 5 3 2 2 1
Marks : 98 99 78 100 98 78 67 100
SGPA : 9.19

o/p seen
8/10/24

Details of for Student 2
USN : 1BM23CS087
Name : Darshan
Credits : 4 4 3 5 3 2 2 1
Marks : 98 48 87 100 98 97 100 61
SGPA : 8.71

CODE:

```
import java.util.Scanner;

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

    // Method to accept student details
    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: ");
        numSubjects = scanner.nextInt();

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

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

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

    // Method to calculate SGPA
```

```
    public double calculateSGPA() {
        int totalCredits = 0;
        int weightedGradePoints = 0;

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

        return (double) weightedGradePoints / totalCredits;
    }

    // Method to calculate grade point based on marks
    private int calculateGradePoint(int marks) {
        if (marks >= 90) return 10;
        else if (marks >= 80) return 9;
        else if (marks >= 70) return 8;
        else if (marks >= 60) return 7;
        else if (marks >= 50) return 6;
        else if (marks >= 40) return 5;
        else return 0; // Fail
    }

    // Method to display student details
    public void displayDetails() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("SGPA: " + calculateSGPA());
    }
}

public class Main {
    public static void main(String[] args) {
        Student student = new Student();
        student.acceptDetails();
        student.displayDetails();
    }
}
```

```
E:\danish_with_java\java_lab\Program 2>javac Main.java

E:\danish_with_java\java_lab\Program 2>java Main
Enter USN: 1BM23CS086
Enter Name: Danish
Enter number of subjects: 2
Enter credits for each subject:
Credits for Subject 1: 4
Credits for Subject 2: 3
Enter marks for each subject:
Marks for Subject 1: 100
Marks for Subject 2: 100
USN: 1BM23CS086
Name: Danish
SGPA: 10.0
```

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.

10/10/2024

Q Create a class Book which Contains 4 members : name, author, price, num_pages. Include a Constructor to set the values for the members. Include methods to set & get the details of objects. Include a toString() method that could display the complete details of book. Develop a Java Program to Create n book objects

Ans

```
class Book
{
    private String name;
    private String author;
    Private int price;
    private int num_pages;
}

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

Public String getName()
{ return name; }
public String getAuthor
{ return author; }
Public int getPrice
{ return price; }
```

```
Public int getNum_pages
{ return num_pages; }

Public toString( )
{
    return "Book_name:"+name "author:"+author
           "\nprice: "+price "\nnum_pages:"+num_pages;
}

public class My_code
{
    public Static void main(String args[])
    {
        Scanner scan = new Scanner();

        System.out.printf("Enter no. of Books");
        int n = scan.nextInt();

        Book[] books = new Book[n];

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

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

            System.out.println("Enter price:");
            int price = scan.nextInt();
```

```
System.out.println("Enter num_pages: ");
int num_pages = scan.nextInt();

        books[i] = new Book(name, author, price, num_pages);
}
        System.out.println("Book details");
for (Book book : books)
{
        System.out.println("Book details");
        System.out.println("----------");
        System.out.println(book.toString());
}
scan.close();
}
```

Seen
execute

Output

```
Enter no. of books: 2
Enter book details of 1:
Enter bookname: Famous Five
Enter author: Dhruva
Enter price: 2999
Enter no. of pages: 1234
Book details:
Book: Famous Five
Author: Dhruva
Price: 2999
Num of pages: 546
Book: Wimpy boy
Author: Danish
Price: 3456
Num of pages: 234
```

Book details

Book name : Famous Five
Author : Dhruva
Price : 2999
Num of pages : 546

Book : Whimpy boy
Author : Danish
Price : 3456
Num of pages : 1234

o/p seen
GL
15/10/24

CODE:

```
import java.util.Scanner;

// Class to represent a Book
class Book {
    // Member variables
    private String name;
    private String author;
    private int price;
    private int num_pages;

    // Constructor to initialize the book details
    public Book(String name, String author, int price, int num_pages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.num_pages = num_pages;
    }

    // Getter
    public String getName() {
        return name;
    }

    public String getAuthor() {
        return author;
    }

    public int getPrice() {
        return price;
    }

    public int getNumPages() {
        return num_pages;
    }

    // toString method to display book details
    //@Override
    public String toString() {
        return "Book Name: " + name + "\nAuthor: " + author + "\nPrice: " + price +
```

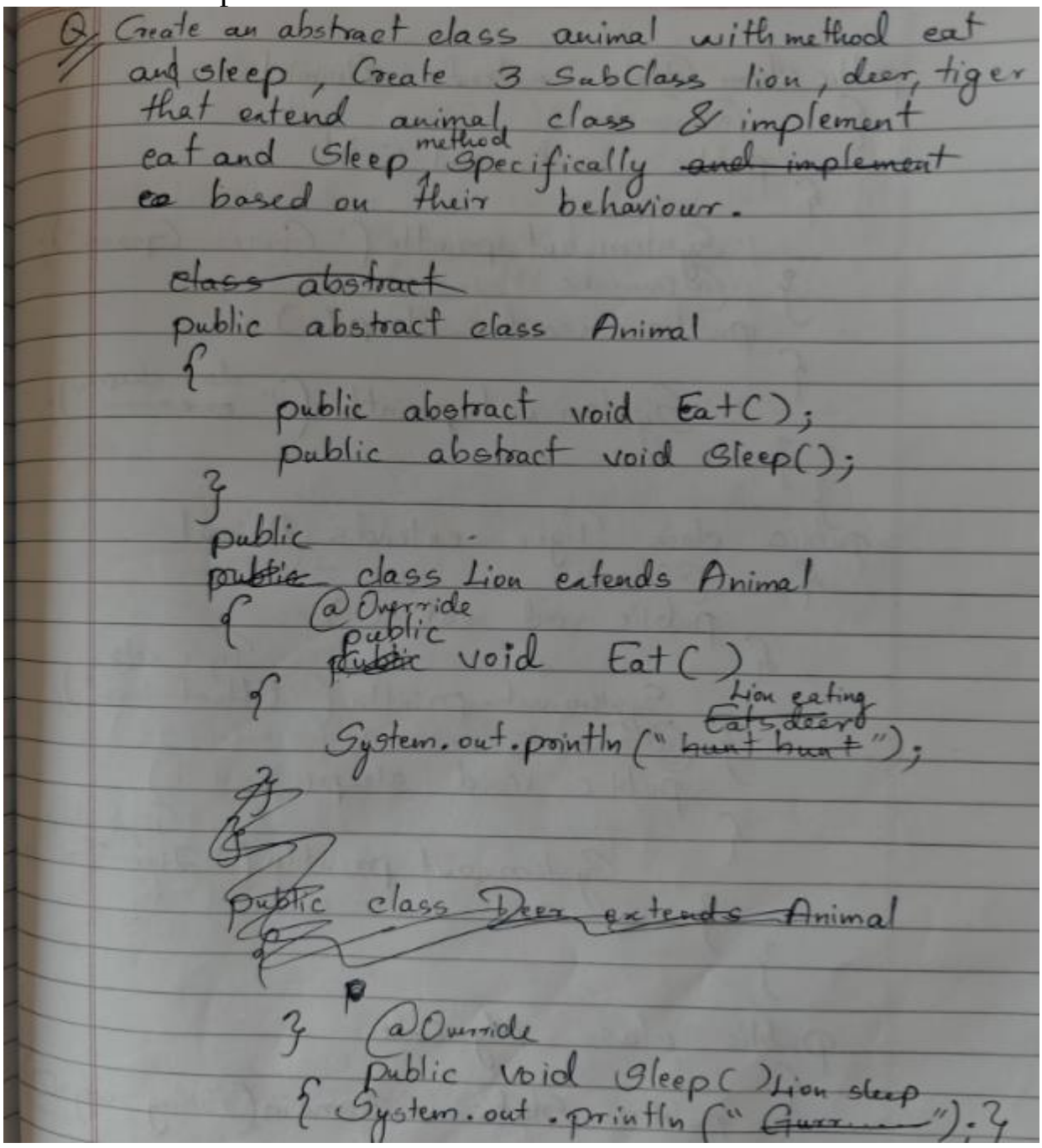
```
"\nNumber of Pages: " + num_pages;
    }
}

public class App {
    public static void main(String[] args) {
        // Scanner object for user input
        Scanner scan = new Scanner(System.in);

        // Asking the user for the number of books
        System.out.print("Enter the number of books: ");
        int n = scan.nextInt();
        scan.nextLine();  // consume newline

        // Array to store Book objects
        Book[] books = new Book[n];

        // Loop to accept details of each book
        for (int i = 0; i < n; i++) {
            System.out.println("Enter details of book " + (i + 1) + ":");

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

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

            System.out.print("Enter price: ");
            int price = scan.nextInt();

            System.out.print("Enter number of pages: ");
            int num_pages = scan.nextInt();
            scan.nextLine();  // consume newline

            // Creating a new Book object and storing it in the array
            books[i] = new Book(name, author, price, num_pages);
        }

        // Displaying all the books
        System.out.println("\nBook Details:");
```

```
        for (Book book : books) {
            System.out.println(book.toString());
            System.out.println("--------------------");
        }

        // Closing the scanner to prevent resource leak
        scan.close();
        System.out.println("Name: Danish\n USN:1BM23CS086");
    }
}
```

```
E:\danish_with_java\java_lab\Program 3>java App
Enter the number of books: 2
Enter details of book 1:
Enter name: Famous five
Enter author: Vatsalya
Enter price: 140
Enter number of pages: 100
Enter details of book 2:
Enter name: How to become rich
Enter author: Danish
Enter price: 1009
Enter number of pages: 499

Book Details:
Book Name: Famous five
Author: Vatsalya
Price: 140
Number of Pages: 100
--------------------
Book Name: How to become rich
Author: Danish
Price: 1009
Number of Pages: 499
--------------------
Name: Danish
 USN:1BM23CS086
```

PROGRAM 4

Create an abstract class Animal with two methods, eat and sleep. Create three subclasses Lion, Deer, and Tiger that extend the Animal class and implement the eat and sleep methods based on their specific behavior. Write a Main class to create objects of these subclasses and demonstrate their behavior by calling the eat and sleep methods. Include your name and USN in the output**.**



```
public class Deer extends Animal
{   @Override
    public void eat()
    {
        System.out.println("Grass Grass");
    }  @Override
    public void sleep()
    {
        System.out.println("deer sleep zzzz...");
    }
}
public class Tiger extends Animal
{   @Override
    public void eat()
    {
        System.out.println("Tiger eats Hunt less");
    }  @Override
    public void sleep()
    {
        System.out.println("Tiger sleep Jizz Jizz");
    }
}

Public class My-Code
{
    public Static void main(String args[])
    {
        Lion lion = new Lion();
        lion.sleep();
        lion.eat();
```

```
Tiger tiger = new Tiger();
        tiger.sleep();
        tiger.eat();
Deer deer = new Deer();
        deer.sleep();
        deer.eat();
}
```

Seen executed.

Output

```
Lion eats meat
Lion sleeps during day
Deer eats grass
Deer sleeps
Tiger eats other animals
Tiger sleeps
```

Seen
GL
22/10/24

CODE:

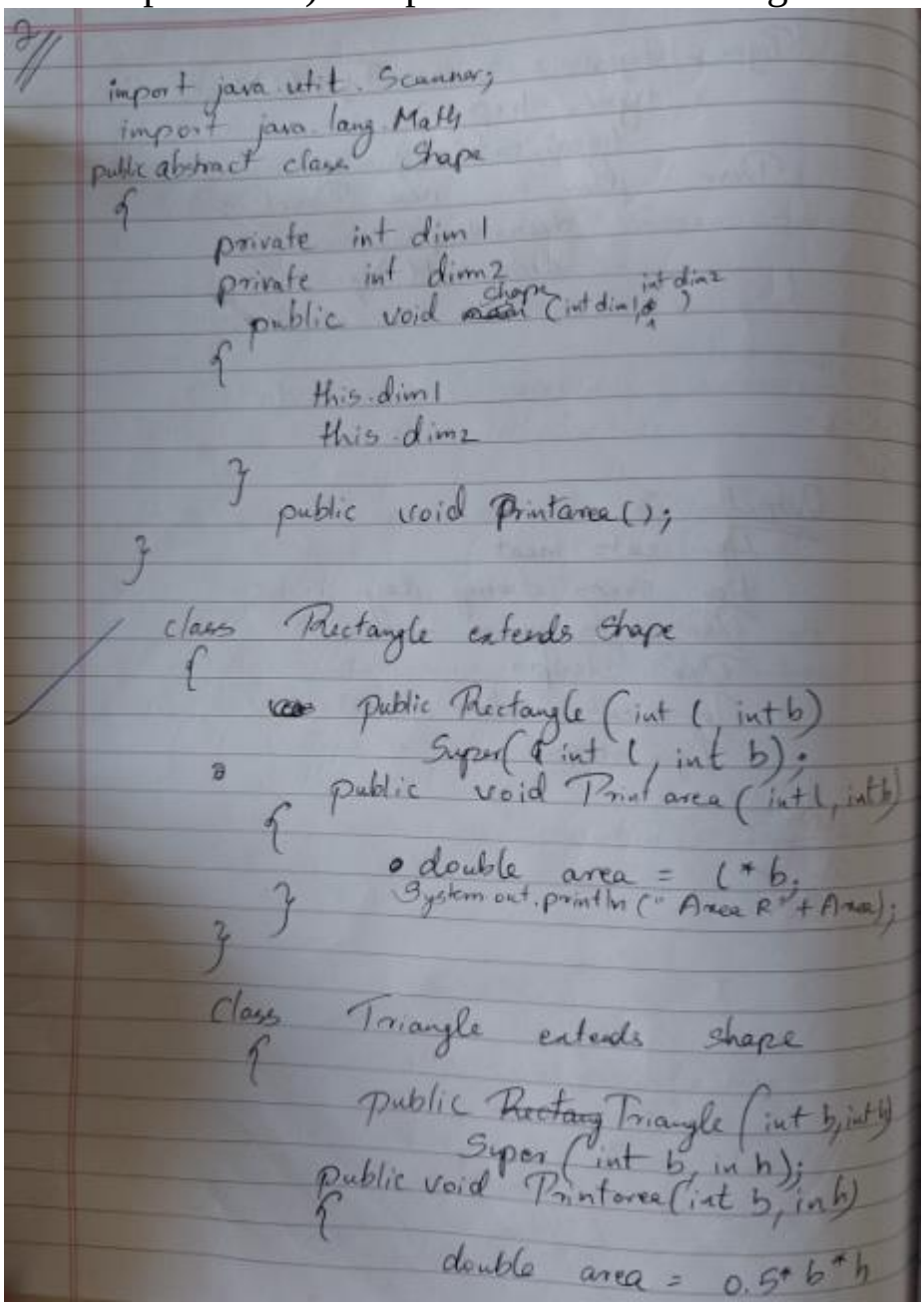
```
abstract class Animal {
    // Abstract method eat
    abstract void eat();

    // Abstract method sleep
    abstract void sleep();
}

// Lion class extending Animal
class Lion extends Animal {
    @Override
    void eat() {
        System.out.println("The Lion hunts and eats meat.");
    }

    @Override
    void sleep() {
        System.out.println("The Lion sleeps for 16-20 hours a day.");
    }
}

// Deer class extending Animal
class Deer extends Animal {
    @Override
    void eat() {
        System.out.println("The Deer grazes on grass, leaves, and shrubs.");
    }

    @Override
    void sleep() {
        System.out.println("The Deer sleeps during the night and is alert during the
day.");
    }
}

// Tiger class extending Animal
class Tiger extends Animal {
    @Override
```

```
    void eat() {
        System.out.println("The Tiger hunts and eats large prey like deer or wild boar.");
    }

    @Override
    void sleep() {
        System.out.println("The Tiger sleeps in the shade during the day and hunts at
night.");
    }
}

// Main class to test the behavior
public class Main {
    public static void main(String[] args) {
        // Creating objects of Lion, Deer, and Tiger
        Animal lion = new Lion();
        Animal deer = new Deer();
        Animal tiger = new Tiger();

        // Testing the behavior of each animal
        lion.eat();
        lion.sleep();

        deer.eat();
        deer.sleep();

        tiger.eat();
        tiger.sleep();
        System.out.println("NAME:Danish \nUSN:1BM23CS086");
    }
}
```

```
E:\danish_with_java\java_lab\Program 4>java Main
The Lion hunts and eats meat.
The Lion sleeps for 16-20 hours a day.
The Deer grazes on grass, leaves, and shrubs.
The Deer sleeps during the night and is alert during the day.
The Tiger hunts and eats large prey like deer or wild boar.
The Tiger sleeps in the shade during the day and hunts at night.
NAME:Danish
USN:1BM23CS086
```

PROGRAM 4(2):

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.



```
System.out.println("Area T" + Area);
}
class Circle extends Shape
{
    public Circle(int r)
        Super(int r);
    public void Printarea(int r)
    {
        double area = 3.14 * r * r;
        System.out.println("Area C" + Area);
    }
public class File_name
{
    public static void main(String args[])
{
        Shape Rectangle r = new Rectangle();
        Triangle t = new Triangle();
        Circle c = new Circle();
        r.Printarea(10,4);
        t.Printarea(10,4);
        c.Printarea(10);
    }
}
```

See

O/P

```
Area of Circle : 12.566370614359172
Area of Rectangle : 8.0
Area of Triangle : 4.0
```

CODE:

```
// Abstract class Shape
abstract class Shape {
    int dimension1;
    int dimension2;

    // Abstract method to calculate and print the area
    abstract void printArea();
}

// Rectangle class extending Shape
class Rectangle extends Shape {
    Rectangle(int length, int breadth) {
        this.dimension1 = length;
        this.dimension2 = breadth;
    }

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

// Triangle class extending Shape
class Triangle extends Shape {
    Triangle(int base, int height) {
        this.dimension1 = base;
        this.dimension2 = height;
    }

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

// Circle class extending Shape
class Circle extends Shape {
```

```
    Circle(int radius) {
        this.dimension1 = radius;
        this.dimension2 = 0; // Not required for a circle
    }

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

// Main class to test the functionality
public class Main {
    public static void main(String[] args) {
        // Create objects for each shape
        Shape rectangle = new Rectangle(10, 5);
        Shape triangle = new Triangle(6, 8);
        Shape circle = new Circle(7);

        // Print areas of the shapes
        rectangle.printArea();
        triangle.printArea();
        circle.printArea();

        System.out.println("NAME: Danish");
        System.out.println("USN: 1BM23CS086");
    }
}
```

```
E:\danish_with_java\java_lab\Program 4\Program 4b>java Main
Area of Rectangle: 50
Area of Triangle: 24.0
Area of Circle: 153.93804002589985
NAME: Danish
USN: 1BM23CS086
```

PROGRAM 5

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

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

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

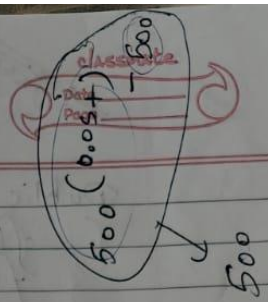
Check for the minimum balance, impose penalty if necessary and update the balance.

28/10/2024

Develop a java ~~Porg~~ program to create a class Bank that maintains 2 types of accounts for Customers, one for Saving other for 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 maintain a min balance if it falls below it service charge will be imposed. Create a class account that stores Customer name, account no. & type of account from this derive the classes cur-acct & Sav-act to make them more specific for req. include the necessary methods in order to achieve following tasks

a. accept deposit from cust & upd the balance
b. display balance
c. Compute & deposit interest
d. permit withdrawal & update balance check for min balance, impose penalty if necessary update the balance.

```
import math.util
~~class Ban~~
import java.util.Scanner;

~~class~~ class BankAccount{
private String[] customerName;
private int[] accountNumber;
private String AccountType
private double[] balance;
}

public BankAccount(String Name,
int AccountNumber, String
accountType, double balance)
{CustomerName[] = this.CustomerName;
accountName[] = this.accountName;
accountType[] = this.accountType;
balance[] = this.balance;
}
```

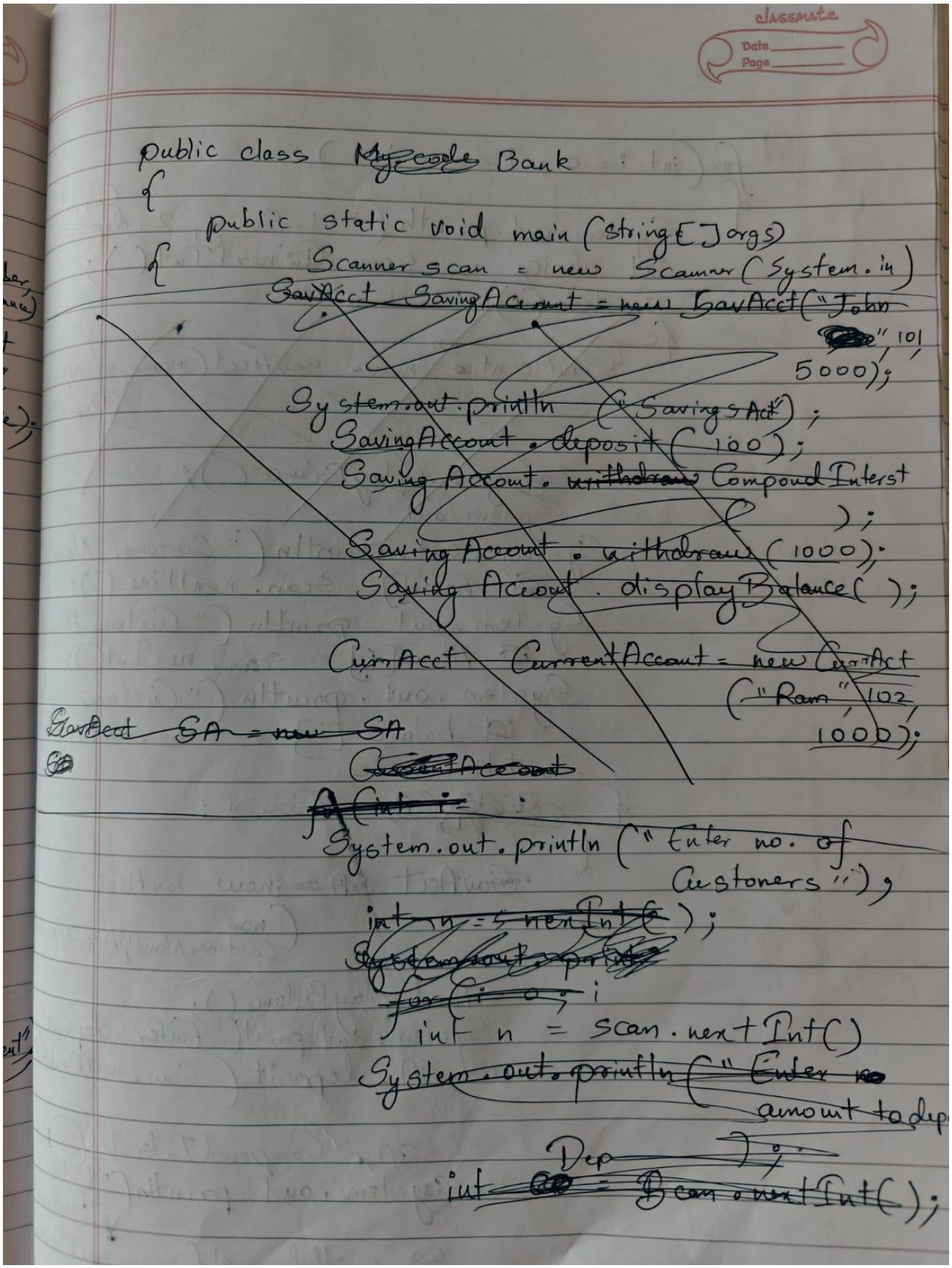


~~Public B~~

```
Public void deposit(double amount)
{
    balance += amount;
    System.out.println("Deposit Succesful! " + balance);
}
public void displayBalance()
{
    System.out.println("Current balance: " + balance);
}
}

Public SavAcct extends BankAccount
{
    final double INTEREST = 0.05;
    public SavAcct (String CustonerName, int accountNumber,
                    double balance)
    { super(CustomerName, accountNumber, "Savings", balance); }

    public void CompoundInterest()
    {
        balance += balance * math.pow(1+INTEREST, 1) - balance;
    }
    public void Withdrawal( double amount)
    {
        if( balance >= amount)
        {
            balance -= amount;
        }
        else
            System.out.println("balance insuffic
        }
```

500

```
Public CurrAcct extends BankAccount
{ final int MIN = 1000;
  int int S_CHARGE = 10;

public CurrAcct(String CustomerName, in AcctNumber
{                                      double bal
   Super (CustomerName, account Number, "Current
                                        Account"
                                        balance);
}

  public void Minbalance ( )
{
     if (balance < MIN)
{       balance -= S_CHARGE;
   System.out.println(" balace is min
So Service Charge imposed, Current
Balance is: + (balance)");
}
{ public void withdrawal
{
   if (balance >= amount)
{
        balance -= amount;
}
   else
        System.out.println(" balace insufficient
}
```


classmate
Date
Page
public class Bank
{
public static void main (String[] args)
{
Scanner scan = new Scanner (System.in)
5000);
System.out.println ("Savings Act");
SavingAccount.deposit (100);
Saving Account. Compound Interst
);
Saving Account. withdraw (1000);
Saving Account. displayBalance ( );
CurrentAccount = new
("Ram", 102,
1000);
System.out.println ("Enter no. of
Customers");
int n = scan.nextInt ();
int n = scan.nextInt()
Enter
amount to dep
int

```
for(int i = 0; i <n ; i++)
{
  System.out.println(" 1. Saving, 2. Current
  int choice = scan.nextInt();
  Switch (i)
  {
    SavAccnt SA = new SavAcct(name, accN,
                                Deposit);
    Savings.displayBalance();
    System.out
  System.out.println(" Customer Name:");
    Custname[i]= scan.nextLine();
  System.out.println(" Customer Account
    Accno[i] = scan.nextInt();
  System.out.println(" Customer bala
    balance[i] = Scan.nextInt();
  Switch choice
  {
    Case 1:
    {
      SavAcct SA = new SavAcct
                   (CustomerName[i]Accno[i]bal
      SA.displayBalance();
      System.out.print(" Enter deposit")
      SA.deposit(Scan.nextDouble());

      SA.CompoundInterst();
      System.out.println(" Enter
                          withdraw")
      SA.withdrawal(scan.nextDouble
```

```
SA.displayBalance();
}
Case 2:
{
CurrAcct CA = new CurrAct(CustomerName[i]
                          Accno[i], balance[i]);
CA.displayBalance();
System.out.println("Enter deposit");
CA.deposit(Scan.nextDouble());
System.out.println("Enter withdrawal");
CA.withdraw(Scan.nextDouble());
CA.displayBalance();
}
Default
{ System.out.println("Invalid Choice");
exit(); }
}
}
Scan.close();
}
```

O/P

Enter no of accounts : 2
Enter Customer name : Danish
Enter account number : 101
Enter initial Balance : 1000
Enter account type (1 for Savings, 2 for Current) : 1

Enter Customer name : Yohitesh
Enter account number : 102
Enter initial Balance : 900

Choose an operation
1. View all accounts
2. Deposit
3. Withdraw
4. Exit
Enter choice : 1

Account Details:
Name : Danish, Acc No : 101, Balance : 1000
Name : yohitesh, Acc No : 102, Balance : 900

Choose an operation:
1. View all accounts
2. Deposit
3. Withdraw
4. Exit
Enter choice : 2

Enter account no to deposit : 101
Enter amount to deposit : 200
Deposit Successful! Updated Balance : 1200

Choose
1. View all accounts
2. Deposit
3. Withdraw
4. Exit
3
Acc no to withdraw : 102
Enter amount to ~~deposit~~ withdraw : 200
Balance below min. update balance : 700
Serive charge imposed

Executed
SL
30/10/24

CODE:

```
import java.util.Scanner;

class BankAccount {
    String customerName;
    int accountNumber;
    String accountType;
    double balance;

    public BankAccount(String customerName, int accountNumber, String
accountType, double balance) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = balance;
    }

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

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

class SavAcct extends BankAccount {
    final double INTEREST_RATE = 0.04; // 4% interest rate

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

    public void computeAndDepositInterest() {
        double interest = balance * Math.pow(1 + INTEREST_RATE, 1) - balance;
        balance += interest;
        System.out.println("Interest added! Updated balance: " + balance);
    }
```

```
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawal successful! Updated balance: " + balance);
        } else {
            System.out.println("Insufficient balance!");
        }
    }
}

class CurAcct extends BankAccount {
    final double MIN_BALANCE = 1000.0;
    final double SERVICE_CHARGE = 100.0;

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

    public void checkMinBalance() {
        if (balance < MIN_BALANCE) {
            balance -= SERVICE_CHARGE;
            System.out.println("Balance below minimum! Service charge imposed.
Updated balance: " + balance);
        }
    }

    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            checkMinBalance();
            System.out.println("Withdrawal successful! Updated balance: " + balance);
        } else {
            System.out.println("Insufficient balance!");
        }
    }
}

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

```
        SavAcct savingsAccount = new SavAcct("John Doe", 101, 5000.0);
        System.out.println("\nSavings Account:");
        savingsAccount.deposit(1000);
        savingsAccount.computeAndDepositInterest();
        savingsAccount.withdraw(2000);
        savingsAccount.displayBalance();


        CurAcct currentAccount = new CurAcct("Jane Doe", 102, 1200.0);
        System.out.println("\nCurrent Account:");
        currentAccount.deposit(500);
        currentAccount.withdraw(100);
        currentAccount.withdraw(2000);
        currentAccount.displayBalance();

        scanner.close();
        System.out.println("Name:Danish \nUSN:1BM23CS086");
    }
}
```

```
E:\danish_with_java\java_lab\Program 5>javac Bank.java

E:\danish_with_java\java_lab\Program 5>java Bank

Savings Account:
Deposit successful! Updated balance: 6000.0
Interest added! Updated balance: 6240.0
Withdrawal successful! Updated balance: 4240.0
Current balance: 4240.0

Current Account:
Deposit successful! Updated balance: 1700.0
Withdrawal successful! Updated balance: 1600.0
Insufficient balance!
Current balance: 1600.0
Name:Danish
USN:1BM23CS086
```

PROGRAM 6

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

Question answer

```
package CIE

public class ~~Personal~~Student
{
    public String USN;
    public String name;
    public int Sem;
}

public Student ~~Personal~~(String USN, String name, int sem)
{
    this.USN = USN
    this.name = name;
    this.Sem = Sem;
}
}

public class Internals extends Studnt
{
    ~~public int[] marks = new int[5];~~

    public Internals (int[] marks)
    {
        this.marks = marks;
    }
}
```

```
package SEE

import CIE.Student

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

    public External( String usn, String name,
      int sem, int[] marks)
    {
        super(usn, name, sem);
        this.marks[] = marks[];
    }
}
```

int sem)

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

public class My_Code
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);
        System.out.print(" Enter no. of Students: ");
```

```
int n = scan.nextInt();
External[] Students = new External[n];
Internals[] internals = new Internals[n];
for(int i=0; i<n; i++)
{
System.out.println("\nStudent " + (i+1));
System.out.println("USN: ");
String usn = scan.next();
System.out.println("Name: ");
String name = scan.next();
System.out.println("Semester: ");
int sem = scan.nextInt();
System.out.print("Internal marks(5 Subjects): ");
int[] intMarks = new int[5];
for(int j=0; j<5; j++)
intMarks[j] = scan.nextInt();
internals[i] = new Internals(intMarks);
```

```
System.out.print("SEE marks (5 subjects): ");
int[] extMarks = new int[5];
for(int j=0; j<5; j++)
    extMarks[j] = Scan.nextInt();
Students[i] = new External(usn, name, sem, extMarks);
}

System.out.println("\nFinal Marks: ");
for(int i=0; i<n; i++)
{
System.out.println(Students[i].name + "(" + Students[i].usn
                                          + ")");
for(int j=0; j<5; j++)
{
int finalMarks = internals[i].marks[j] + (Students[i].marks[j]
                                          / 2);
System.out.println("Course" + (j+1) + ":" + finalMarks);
}
System.out.println();
}
Scan.close();
}
}
```

O/P

Enter no. of Students : 2
Enter ~~USN~~ Name : YO-Hitesh
Enter ~~Name~~ USN : 1BM23CS085
Enter 5 Internal marks
38
50
49
49
50
Enter 5 SEE Marks
50
49
50
41
50
Enter ~~USN~~ Name : Danish
Enter USN : 1BM23CS086
Enter 5 Internal marks
50
50
50
50
50
Enter ~~5~~ External marks
50
50
50
50
50

Final Marks Student :
USN : 1BM23CS085
Name : YO-Hitesh
Semster : 3
Final Marks in 5
Courses : 88 99 90
100
USN : 1BM23CS086
Name : Danish
Semster : 3
Final Marks in 5
Courses : 100 100 100
100

CODE:

Personal.java

```
package CIE;

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

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

}
```

Internals.java

```
package CIE;

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


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

Externals.java

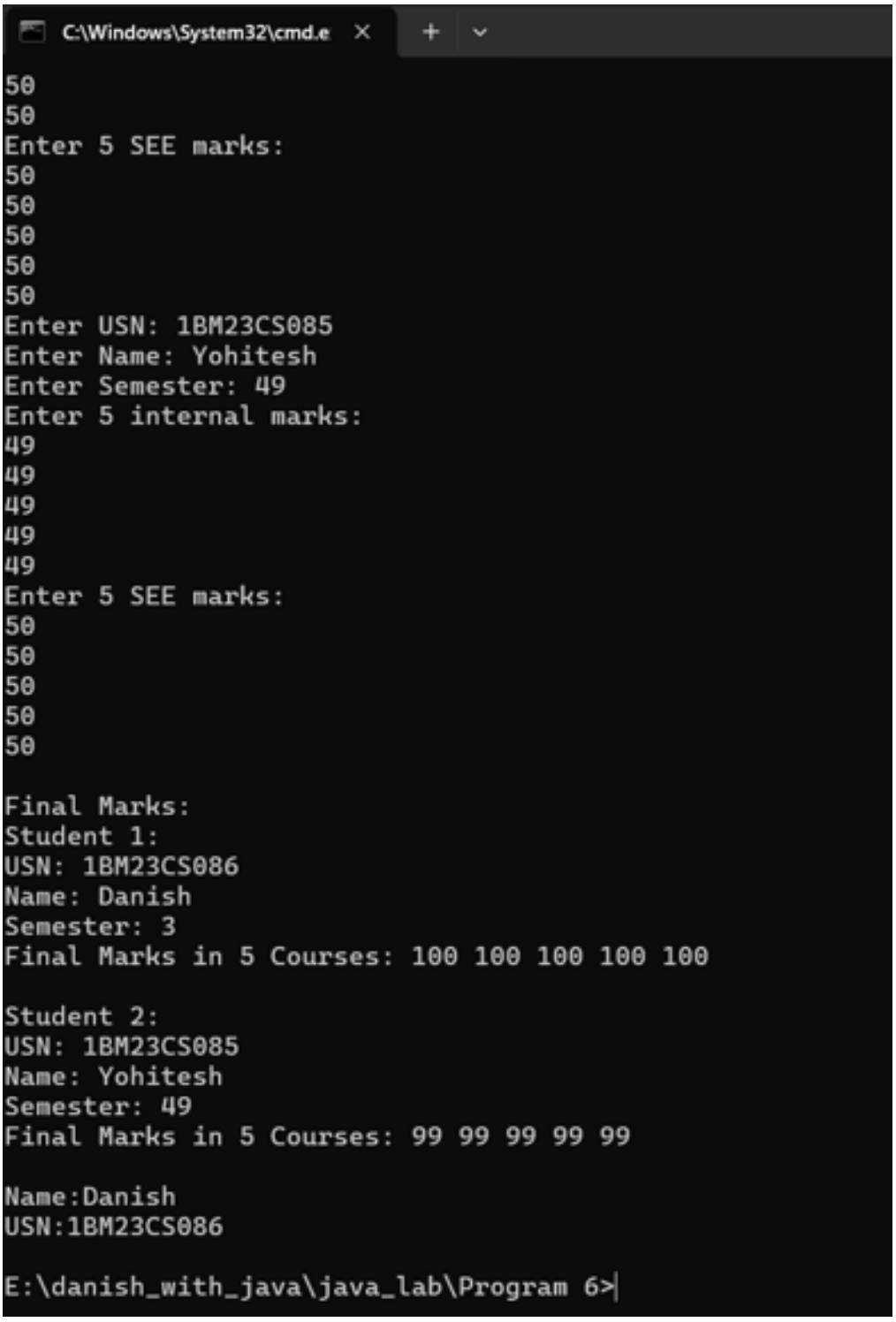
```
package SEE;

import CIE.Personal;

public class External extends Personal {
    public int[] seeMarks;

    // Constructor to initialize the SEE marks
    public External(String usn, String name, int sem, int[] seeMarks) {
```

```
        super(usn, name, sem);
        this.seeMarks = seeMarks;
    }
}
```

Marks.java

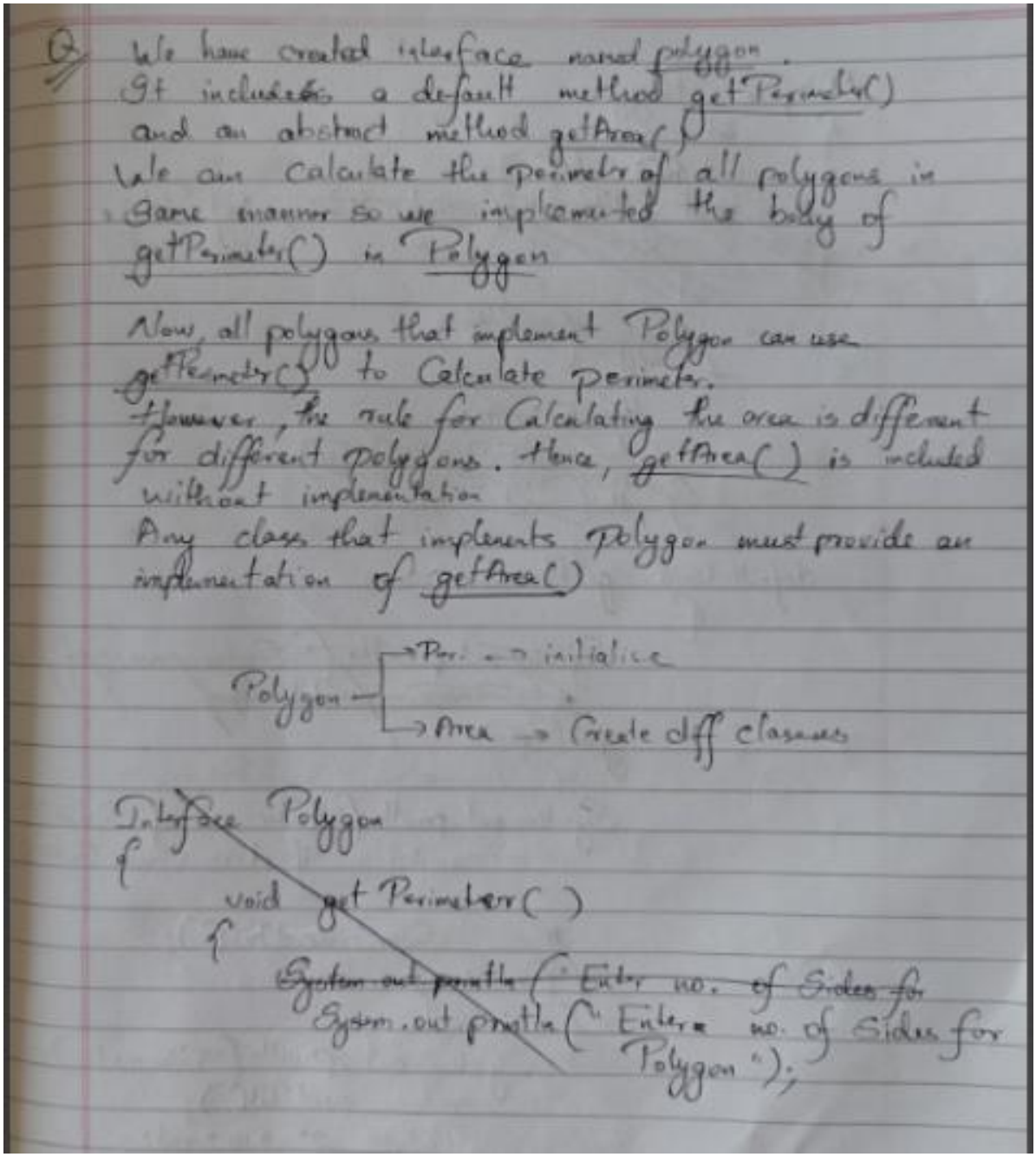
```
import CIE.Internals;
import SEE.External;
import java.util.Scanner;

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

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

        Internals[] internalStudents = new Internals[n];
        External[] externalStudents = new External[n];

        for (int i = 0; i < n; i++) {
            sc.nextLine(); // Clear the buffer

            System.out.print("Enter USN: ");
            String usn = sc.nextLine();
            System.out.print("Enter Name: ");
            String name = sc.nextLine();
            System.out.print("Enter Semester: ");
            int sem = sc.nextInt();

            int[] internalMarks = new int[5]; // Assuming 5 internal marks for simplicity
            System.out.println("Enter 5 internal marks:");
            for (int j = 0; j < 5; j++) {
                internalMarks[j] = sc.nextInt();
            }

            internalStudents[i] = new Internals(usn, name, sem, internalMarks);

            int[] seeMarks = new int[5]; // Assuming 5 SEE marks
            System.out.println("Enter 5 SEE marks:");
```

```
            for (int j = 0; j < 5; j++) {
                seeMarks[j] = sc.nextInt();
            }

            externalStudents[i] = new External(usn, name, sem, seeMarks);
        }

        System.out.println("\nFinal Marks:");

        for (int i = 0; i < n; i++) {
            System.out.println("Student " + (i + 1) + ":");
            System.out.println("USN: " + internalStudents[i].USN);
            System.out.println("Name: " + internalStudents[i].name);
            System.out.println("Semester: " + internalStudents[i].sem);

            System.out.print("Final Marks in 5 Courses: ");
            for (int j = 0; j < 5; j++) {
                // Calculate final mark for each course
                int finalMark = internalStudents[i].internalMarks[j] +
externalStudents[i].seeMarks[j];
                System.out.print(finalMark + " ");
            }
            System.out.println("\n");
        }

        sc.close();
        System.out.println("Name:Danish \nUSN:1BM23CS086");
    }
}
```


C:\Windows\System32\cmd.e
50
50
Enter 5 SEE marks:
50
50
50
50
50
Enter USN: 1BM23CS085
Enter Name: Yohitesh
Enter Semester: 49
Enter 5 internal marks:
49
49
49
49
49
Enter 5 SEE marks:
50
50
50
50
50
Final Marks:
Student 1:
USN: 1BM23CS086
Name: Danish
Semester: 3
Final Marks in 5 Courses: 100 100 100 100 100
Student 2:
USN: 1BM23CS085
Name: Yohitesh
Semester: 49
Final Marks in 5 Courses: 99 99 99 99 99
Name:Danish
USN:1BM23CS086
E:\danish_with_java\java_lab\Program 6>

PROGRAM 7

Write a Java program to create an interface Polygon with methods getArea() and getPerimeter(). Implement two classes, Rectangle and Circle, to calculate the area and perimeter of a rectangle and a circle. Use a Main class to test these implementations and display the results along with your name and USN.



Interface Polygon
{
int n;
int Side[n];
void getPerimeter(int Peri, int Peri;
int radi;
}

int
Polygon(int Peri)
{
this.n = n;
this.Side[n] = Side[n];
this.Peri = Peri;
this.radi = radi;
}

default void getPerimeter
{
System.out.println(" Enter no. of Sides");
Scan.nextInt();

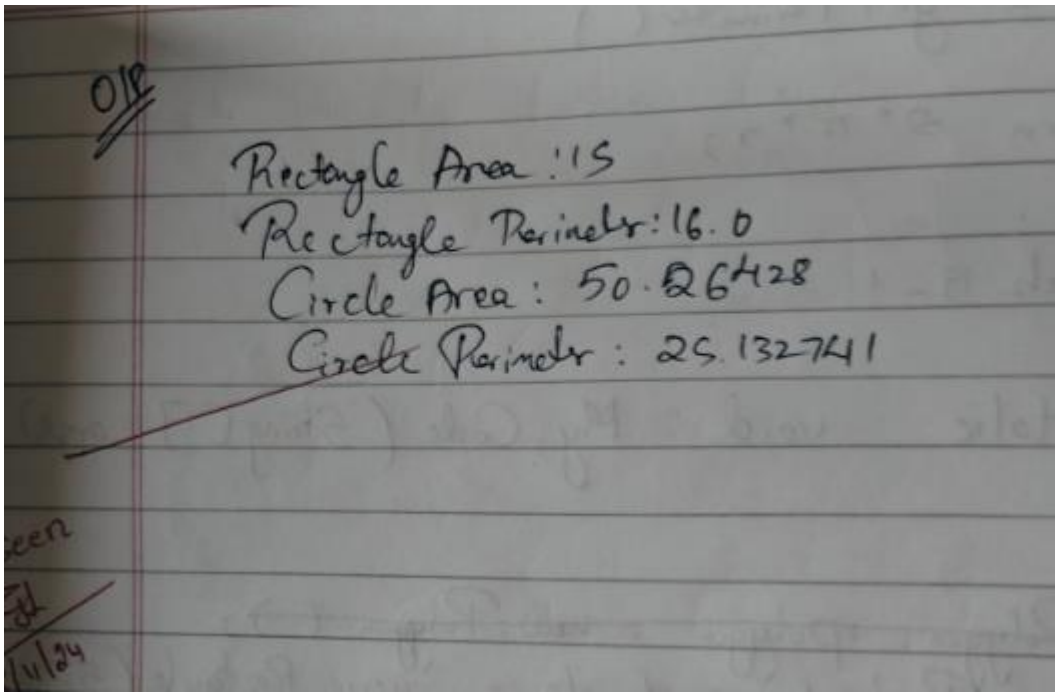
System.out.println(" Enter no. of Sides"(
if it is Circle Enter 0));

n = Scan.nextInt();
if(n == 0)
{
System.out.println(" Enter radi");
radi = nextInt();
Peri = 2* 3.14 * radi;
}

```
else
{
    for(i=0; i<n; i++)
    {
        Peri = 0;
        System.out.println("Enter Side"+[i+1]
        Side[i] = scan.nextInt()
        Peri = Peri + Side[i];
    }
}

abstract getArea();
{
}

class Rectangle implements Polygon
{
    abstract getArea()
    {
        private double length;
        private double width;

        public Rectangle(double length, double width)
        {
            this.length = length;
            this.width = width;
        }
```

```
@Override
public double getArea()
{
    return length * width;
}

}

public class Circle implements
    public double getPerimeter()
    {
        return 2* l* w;
    }
}

public class Circle implements Polygon
{
    private int radi;
{public Circle (int radi)
    this.radi = radi
}
{public double getArea()
    return 3.14*r*r;
}
```

```
public double getPerimeter()
{
    return 2*π*r;
}
}
public class lab_19_11
{
public static void My_Code(String[] args)
{
    Polygon polygon = new Polygon();
    Rectangle rectangle = new Rectangle(5,3);
    System.out.println("Rect area" + rectangle.getArea());
    System.out.println("Rect Peri" + rectangle.getPerimeter());
    Circle circle = new Circle(4);
    System.out.println("Circle Area:" + Circle.
                                    getArea());
    System.out.println("Circle Perimeter:"
                    + Circle.getPerimeter());
}
}
```



CODE:

Polygon.java

```
// Polygon.java
public interface Polygon {
    default double getPerimeter() {
        return 0.0;
    }

    double getArea();
}
```

Circle.java

```
// Circle.java
public class Circle implements Polygon {
    private double radius;

    public Circle(double radius) {
        this.radius = radius;
    }

    @Override
    public double getArea() {
        return Math.PI * radius * radius; // Area of circle: π * r^2
```

```
    }

    @Override
    public double getPerimeter() {
        return 2 * Math.PI * radius; // Perimeter of circle: 2 * π * r
    }
}
```

Rectangle.java

```
// Rectangle.java
public class Rectangle implements Polygon {
    private double length;
    private double width;

    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }

    @Override
    public double getArea() {
        return length * width; // Area of rectangle: length * width
    }

    @Override
    public double getPerimeter() {
        return 2 * (length + width); // Perimeter of rectangle: 2 * (length + width)
    }
}
```

Main.java

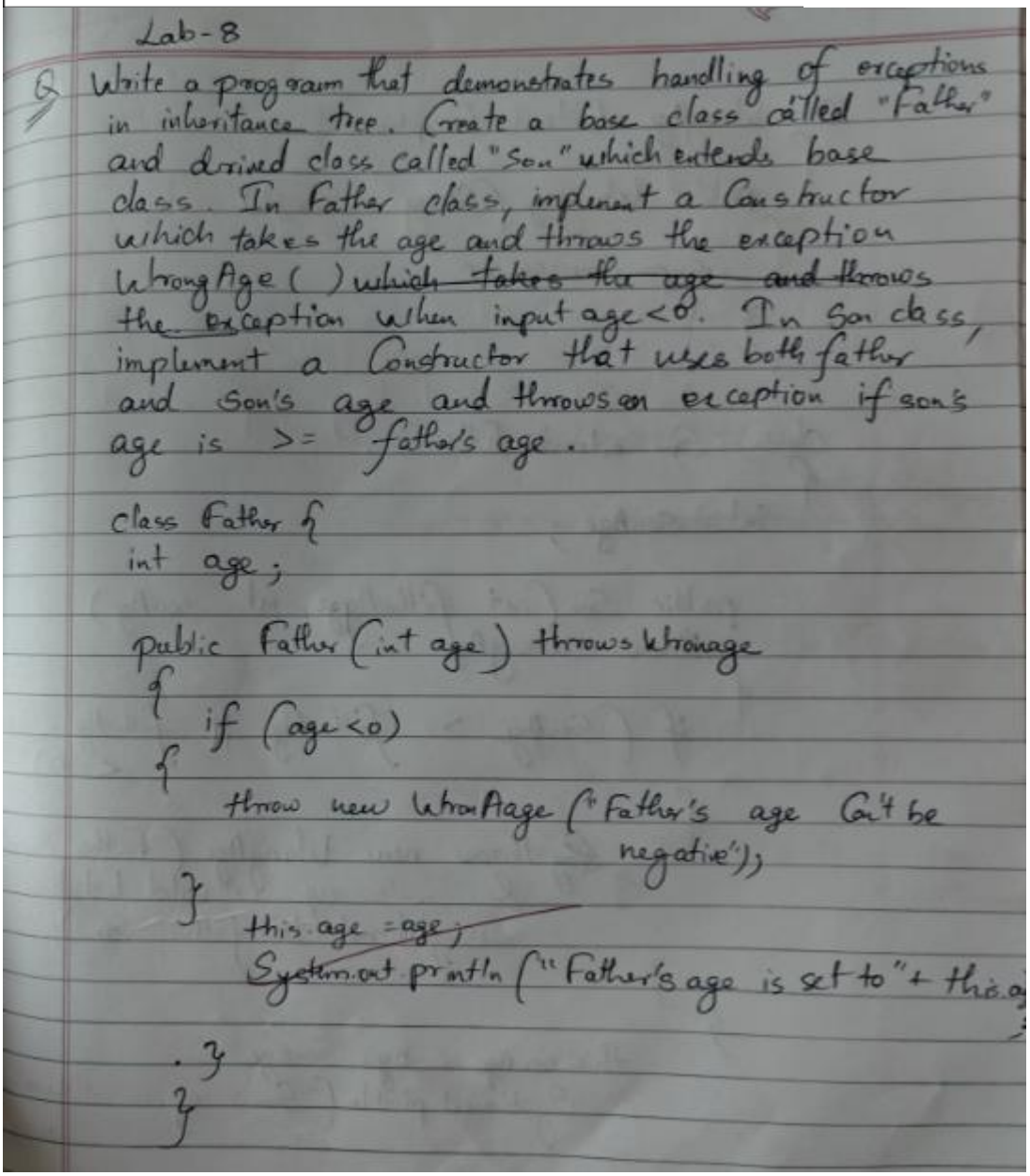
```
// Main.java
public class Main {
    public static void main(String[] args) {
        Rectangle rectangle = new Rectangle(5, 3);
        System.out.println("Rectangle Area: " + rectangle.getArea());
        System.out.println("Rectangle Perimeter: " + rectangle.getPerimeter());

        Circle circle = new Circle(4);
        System.out.println("Circle Area: " + circle.getArea());
```

```
        System.out.println("Circle Perimeter: " + circle.getPerimeter());

        System.out.println("Name: Danish \nUSN: 1BM23CS086");
    }
}
```

```
E:\danish_with_java\java_lab\Program 7>javac Polygon.java

E:\danish_with_java\java_lab\Program 7>javac Rectangle.java

E:\danish_with_java\java_lab\Program 7>javac Circle.java

E:\danish_with_java\java_lab\Program 7>javac Main.java

E:\danish_with_java\java_lab\Program 7>java Main
Rectangle Area: 15.0
Rectangle Perimeter: 16.0
Circle Area: 50.26548245743669
Circle Perimeter: 25.132741228718345
Name: Danish
USN: 1BM23CS086
```

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 WrongAge( ) when the input age<0. In Son class, implement a constructor that uses both father and son's age and throws an exception if son's age is >=father's age.



Lab-8

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

```
class Father {
 int age;

 public Father (int age) throws WrongAge
 {
   if (age<0)
   {
     throw new WrongAge ("Father's age Can't be
                          negative");
   }
     this.age = age;
     System.out.println ("Father's age is set to" + this.a
   }
}
```

```
class WrongAge Extends Exception
{
    public Wrongage(String message)
    {
        super(message);
    }
}

class Son extends father
{
    int SonAge;

    public Son(int fatherAgge, int SonAge)
      throws WrongAge
    {
        if (SonAge >= fatherAge || fatherAge - SonAge
                                        <= 18)
        {
            throw new WrongAge("Father's age
                & Son's age is valid but they
                might not be father Son");
        }
    }

        this.Soonage = agge Sonage
        System.out.println("Son's age is set to→"+
                                        this.Sonage);
}
```

```
public class Main_Exception
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);
        System.out.println("Enter fathers age");
        int Fage = nextInt();
        System.out.println("Enter Son's age");
        int Sage = nextInt();

        Son & soni = new Son(Fage, Sage);

        try
        {
            Son son1 = new Son(Fage, Sage);
        }
        Catch(Exception e)
        {
            System.out.println("Exception -> " + e.getMessage
        }
    }
}
```

Wrong Age

O/P

Enter father's age
40
Enter Son's age
30
Father's age set to → 40
Exception Caught → Father's age and Son's age is valid but they might not be father and Son.

Seen
26/11/24

CODE:

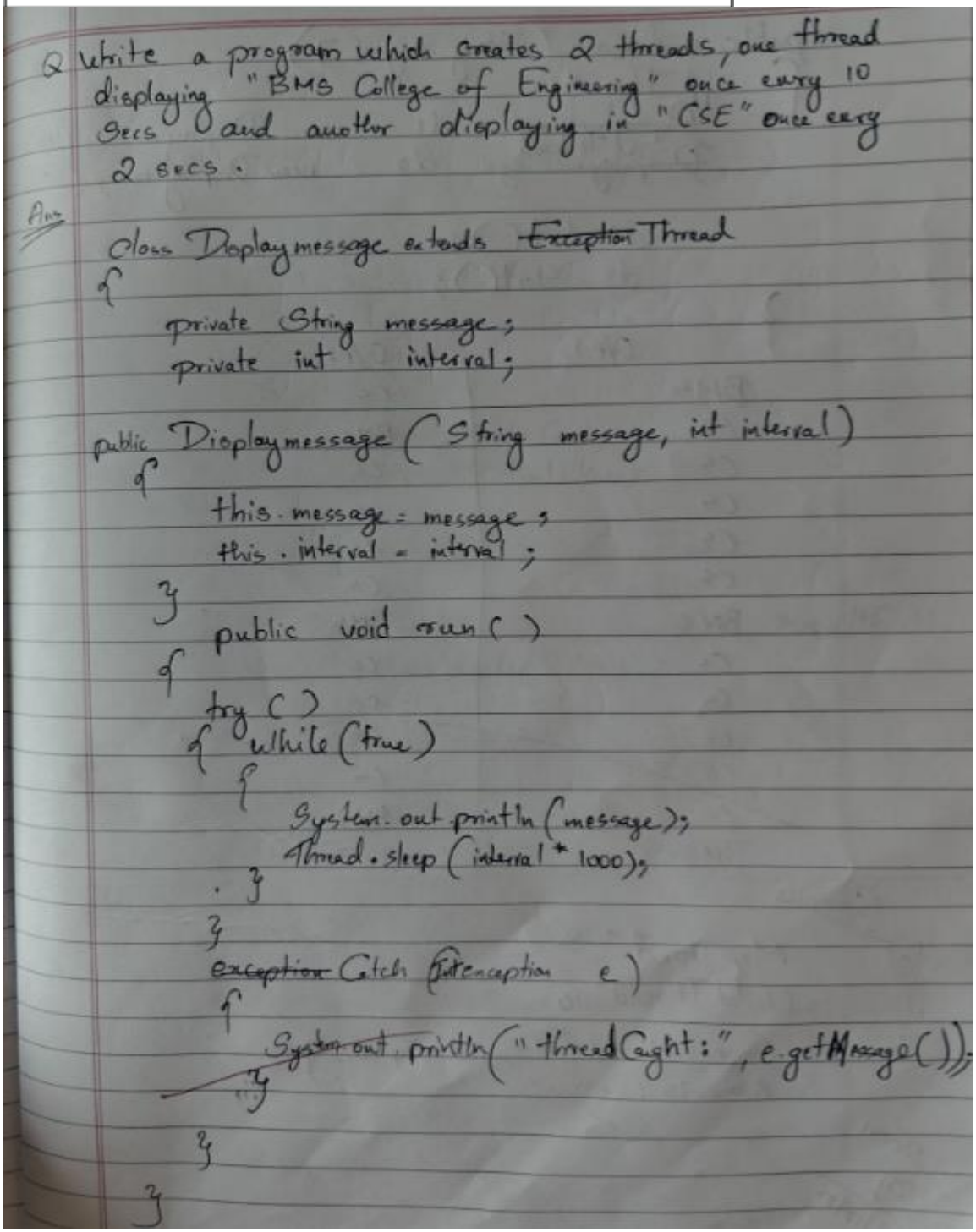
```
import java.util.Scanner;

// Custom exception class
class WrongAge extends ArithmeticException {
    public WrongAge(String msg) {
        super(msg);
    }
}

// Father class to handle father's age and validation
class Father {
    int father_age;

    // Constructor to validate father's age
    public Father(int father_age) throws WrongAge {
        if (father_age <= 0) {
            throw new WrongAge("Father's age can't be negative or zero.");
        } else if (father_age < 18) {
            throw new WrongAge("Father's age is valid, but he can't be a father at such a
young age.");
        }
        this.father_age = father_age;
        System.out.println("Father's age is set: " + father_age);
    }
}

// Son class that inherits from Father class and checks son's age
class Son extends Father {
    int son_age;

    // Constructor to validate son's age and age gap with father
    public Son(int son_age, int father_age) throws WrongAge {
        super(father_age);

        if (son_age <= 0) {
            throw new WrongAge("Son's age can't be negative or zero.");
        } else if (father_age - son_age < 18) {
            throw new WrongAge("The age gap between father and son should be greater
than 18 years.");
```

```
        }

        this.son_age = son_age;
        System.out.println("Son's age is set: " + son_age);
    }
}

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

        // Input for father's age
        System.out.println("Enter the age of father:");
        int fage = scan.nextInt();

        // Input for son's age
        System.out.println("Enter the age of son:");
        int sage = scan.nextInt();

        try {
            // Create Son object which invokes Father constructor
            Son son = new Son(sage, fage);
        } catch (WrongAge e) {
            // Catch the exception and print the message
            System.out.println("Exception caught -> " + e.getMessage());
        }

        System.out.println("Name: Danish\nUSN:1BM23CS086");
    }
}
```

```
E:\danish_with_java\java_lab\Program 2>javac Main_Exception.java

E:\danish_with_java\java_lab\Program 2>java Main_Exception
Enter the age of father:
40
Enter the age of son:
20
Father's age is set: 40
Son's age is set: 20
Name: Danish
USN:1BM23CS086
```

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.



Q Write a program which creates 2 threads, one thread displaying "BMS College of Engineering" once every 10 secs and another displaying in "CSE" once every 2 secs.

Ans

```
class Displaymessage extends Exception Thread
{
    private String message;
    private int interval;

public Displaymessage (String message, int interval)
{
    this.message = message;
    this.interval = interval;
}
    public void run()
{
  try()
  { while(true)
    {
        System.out.println(message);
        Thread.sleep(interval * 1000);
    }
  }
  exception Catch (Interception e)
  {
      System.out.println("threadCaught:", e.getMessage());
  }
}
```

```
public void main (String args[])
{
    Thread
    Displaymessage d1 = new Displaymessage("BMS");
    Displaymessage d2 = new Displaymessage("CS");

    d1.start();
    d2.start();
}
```

O/P 1	O/P 2
BMS	CS
CS	BMS
CS	CS
CS	CS
CS	CS
CS	CS
BMS	BMS
CS	CS
CS	CS
CS	CS
CS	CS
CS	CS
BMS	BMS

Sample

1. Main Thread * 10
   Child Thread * 10
2. Current thread [#1, main 5, main]
   Name is : main

Soon

03/12/24

CODE:

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

    public DisplayMessage(String message, int interval) {
        this.message = message;
        this.interval = interval;
    }

    public void run() {
        try {
            while (true) {
                System.out.println(message);
                Thread.sleep(interval * 1000); // Convert seconds to milliseconds
            }
        } catch (InterruptedException e) {
            System.out.println("Thread interrupted: " + e.getMessage());
        }
    }
}

public class MultithreadingDemo {
    public static void main(String[] args) {
        Thread thread1 = new DisplayMessage("BMS College of Engineering", 10);
        Thread thread2 = new DisplayMessage("CSE", 2);
        System.out.println("NAME:Danish \nUSN:1BM23CS086");
        thread1.start();
        thread2.start();
    }
}
```

```
E:\danish_with_java\java_lab\Program 9>java MultithreadingDemo
NAME:Danish
USN:1BM23CS086
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
```

PROGRAM 10

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.

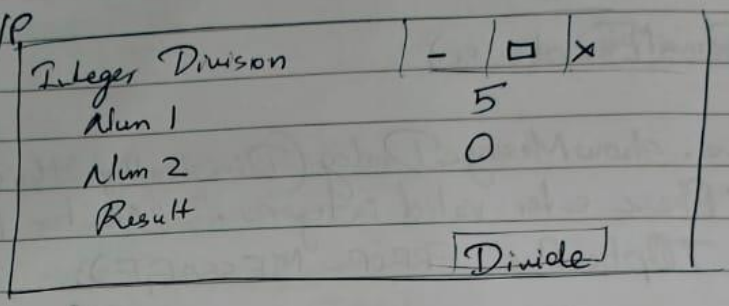
Question 2

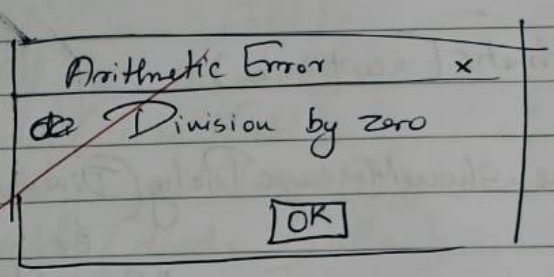
Write a program that creates a user interface to perform int division. The user enters 2 no.s in text fields, Num1 and Num2. The division of Num1 & Num2 is displayed in Result field when the divide button is clicked. If num1 and/or Num2 not int, shd throw NumberFormatException. If Num2 were 0 Program would throw an Arithmetic Exception Display the exception in message dialog box

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

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

    public DivisionApp()
    {
        setTitle("Integer Division");
        setSize(300, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new GridLayout(4, 2));

        JLabel num1Label = new JLabel("Num 1:");
        num1Field = new JTextField();
        JLabel num2Label = new JLabel("Num 2:");
        num2Field = new JTextField();
        JLabel resultLabel = new JLabel("Result:");
        resultField = new JTextField();
```

```
resultField.setEditable(false);
divideButton = new JButton("Divide");

add(num1Label);
add(num2Field);
add(num2Label);
add(num2Field);
add(ResultLabel);
add(resultField);
add(new JLabel());
add(divideButton);

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));
}
```

O/P

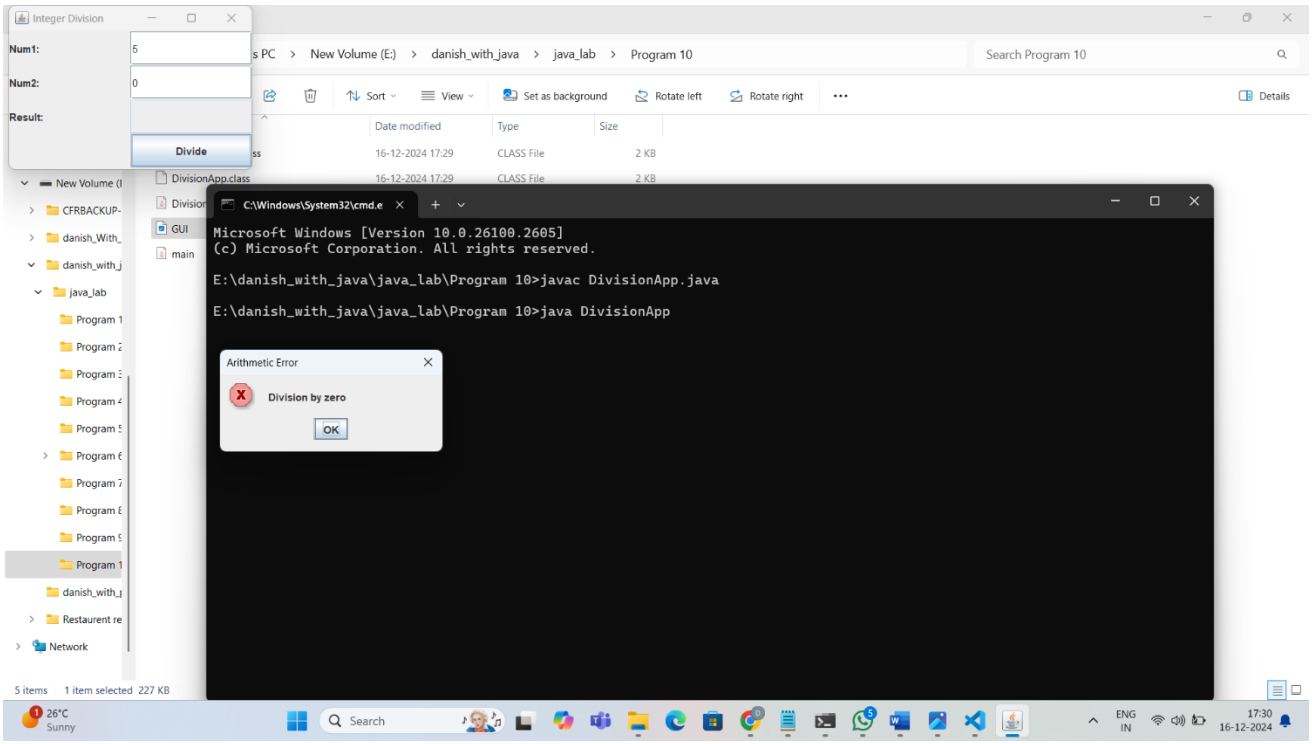

Integer Division
Num 1
5
Num 2
0
Result
Divide


Arithmetic Error
Division by zero
OK

CODE:import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

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

    public DivisionApp() {
        // Setting up the frame
        setTitle("Integer Division");
        setSize(300, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new GridLayout(4, 2));

        // Creating components
        JLabel num1Label = new JLabel("Num1:");
        num1Field = new JTextField();
        JLabel num2Label = new JLabel("Num2:");
        num2Field = new JTextField();
        JLabel resultLabel = new JLabel("Result:");
        resultField = new JTextField();
        resultField.setEditable(false);
        divideButton = new JButton("Divide");

        // Adding components to the frame
        add(num1Label);
        add(num1Field);
        add(num2Label);
        add(num2Field);
        add(resultLabel);
        add(resultField);
        add(new JLabel()); // Empty cell
        add(divideButton);

        // Adding action listener to the button
        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(DivisionApp.this,
                "Please enter valid integers.", "Number Format Error",
                JOptionPane.ERROR_MESSAGE);
        } catch (ArithmeticException ex) {
            JOptionPane.showMessageDialog(DivisionApp.this,
                ex.getMessage(), "Arithmetic Error",
                JOptionPane.ERROR_MESSAGE);
        }
      }
    });
  }

  public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
      DivisionApp app = new DivisionApp();
      app.setVisible(true);
    });
  }
}
```


Integer Division
Num1:
5
Num2:
0
Result:
Divide
Program 10
Search Program 10
Sort
View
Set as background
Rotate left
Rotate right
Details
Date modified
Type
Size
16-12-2024 17:29
CLASS File
2 KB
DivisionApp.class
GUI
main
C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.26100.2605]
(c) Microsoft Corporation. All rights reserved.
E:\danish_with_java\java_lab\Program 10>javac DivisionApp.java
E:\danish_with_java\java_lab\Program 10>java DivisionApp
Arithmetic Error
Division by zero
OK
5 items
1 item selected 227 KB
26°C
Sunny
Search
ENG
IN
17:30
16-12-2024