

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

SANCHIT MEHTA (1BM23CS299)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING



B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019 Sep

2024-Jan 2025

B. M. S. College of Engineering,
Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering



CERTIFICATE

This is to certify that the Lab work entitled “**OBJECT ORIENTED JAVA PROGRAMMING**” carried out by **SANCHIT MEHTA(1BM23CS299)**, who is bonafide student of **B. M. S. College of Engineering**. It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Object-Oriented Java Programming Lab - (23CS3PCOOJ)** work prescribed for the said degree.

Dr. Nandhini Vineeth

Associate Professor,
Department of CSE,
BMSCE, Bengaluru

Dr. Kavitha Sooda

Professor and Head,
Department of CSE
BMSCE, Bengaluru

INDEX

Sl. No.	Date	Experiment Title	Page No.
1	20/09/24	Quadratic Equation	1
2	03/10/24	SGPA Calculator	5
3	19/10/24	Book Details	10
4	24/10/24	Abstract Shape	17
5	07/10/24	Bank Operations	21
6	14/11/24	CIE/SEE Packages	30
7	21/11/24	Father-Son Age Exception	39
8	05/12/24	Multi-Threading	43
9	12/12/24	Custom division using awt	46
10	19/12/24	Demonstrate Inter process Communication and deadlock	49

LABORATORY 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 discriminant $b^2 - 4ac$ is negative, display a message stating that there are no real solutions

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 quadratic formula. If discriminant is negative, display there are no real solutions.
26/09/2020

Quadratic Equation

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

```
public class QuadraticSolver {
```

```
    public static void main(String[] args) {
```

```
        System.out.print("Enter coefficient a:");
```

```
        double a = scanner.nextDouble();
```

```
        System.out.print("Enter coefficient b:");
```

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

```
        System.out.print("Enter coefficient c:");
```

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

```
        double discriminant = b*b - 4*a*c;
```

```
        if (discriminant > 0) {
```

```
            double root1 = (-b + Math.sqrt(discriminant)) / (2*a);
```

```
            double root2 = (-b + Math.sqrt(discriminant)) / (2*a);
```

```
            System.out.print("Roots are real and different");
```

```
        }
```

```
        else if (discriminant == 0) {
```

```
            double root = -b / (2*a);
```

```
            System.out.print("Roots are real and same, root");
```

```
        } else {
```

```
            double real part = -b / (2*a);
```

```
            double imaginary part = Math.sqrt(-discriminant) / (2*a);
```

```
            System.out.print("Roots are complex and different\nRoot 1 = %.2f + %.2fi\nRoot 2 = %.2f - %.2fi\n");
```

```

    }
    Scanner.close()
}

```

Output:-> Enter Coefficient a: 12
 Enter Coefficient b: 13
 Enter Coefficient c: 14
 Roots are complex and different
 Root 1 = $-0.54 + 0.93i$
 Root 2 = $-0.54 - 0.93i$

Enter coefficient a: 2
 Enter coefficient b: 1
 Enter coefficient c: 0
 Roots are real and distinct
 Root 1 = 0.00
 Root 2 = -0.50

Enter coefficient a: 1
 Enter coefficient b: 2
 Enter coefficient c: 1
 Roots are real and same
 Root = -1.0

26/9/24

Code:

```
import java.util.Scanner;

public class lab1 {

    public static void main(String[] args) {

        try (Scanner scanner = new Scanner(System.in)) {

            System.out.print("Enter coefficient a: ");

            double a = scanner.nextDouble();

            System.out.print("Enter coefficient b: ");

            double b = scanner.nextDouble();

            System.out.print("Enter coefficient c: ");

            double c = scanner.nextDouble();

            if (a == 0) {

                System.out.println("Coefficient a cannot be zero for a quadratic equation.");

                return;

            }

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

            if (discriminant > 0) {

                double sqrtDiscriminant = Math.sqrt(discriminant);

                double root1 = (-b + sqrtDiscriminant) / (2 * a);

                double root2 = (-b - sqrtDiscriminant) / (2 * a);

                System.out.println("The roots are real and different.");

                System.out.println("Root 1: " + root1);

                System.out.println("Root 2: " + root2);

            } else if (discriminant == 0) {

                double root = -b / (2 * a);
```

```

        System.out.println("The root is real and repeated.");

        System.out.println("Root: " + root);

    } else {

        System.out.println("There are no real solutions.");

    }

}

}

}

```

OUTPUT

```

Microsoft Windows [Version 10.0.22631.4460]
(c) Microsoft Corporation. All rights reserved.

C:\Users\sanch>cd C:\Users\sanch\Desktop\java
C:\Users\sanch\Desktop\java>javac lab1.java
C:\Users\sanch\Desktop\java>java lab1
Enter coefficient a: 12
Enter coefficient b: 13
Enter coefficient c: 14
There are no real solutions.

C:\Users\sanch\Desktop\java>java lab1
Enter coefficient a: 2
Enter coefficient b: 1
Enter coefficient c: 0
The roots are real and different.
Root 1: 0.0
Root 2: -0.5

C:\Users\sanch\Desktop\java>java lab1
Enter coefficient a: 1
Enter coefficient b: 2
Enter coefficient c: 1
The root is real and repeated.
Root: -1.0

```


LABORATORY PROGRAM – 2

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

30/10/2024



Q. 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.

```
import java.util.Scanner
```

```
Class Student {
```

```
    String usn;  
    String name;  
    int[] credits;  
    int marks;
```

```
    void acceptingDetailsOfStudents() {
```

```
        Scanner sc = new Scanner(System.in);
```

```
        System.out.println("Enter USN:");
```

```
        usn = sc.nextLine();
```

```
        System.out.print("Enter name:");
```

```
        name = sc.nextLine();
```

```
        System.out.print("Enter no. of subjects:");
```

```
        int n = sc.nextInt();
```

```
        credits = new int[n];
```

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

```
            System.out.print("Enter credits for sub "+  
                (i+1) + ":");
```

```
            marks[i] = sc.nextInt();
```

```
        }
```

```
    }
```

```
    void displayDetails() {
```

```
        System.out.println("USN: " + usn);
```

```
        usn = sc.nextLine();
```

```
        System.out.print("Name: " + name);
```

```
        System.out.print("Credit and Marks:");
```



```

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

```

```

double calculate SGPA() {
    int totalCredits = 0;
    double weightedSum = 0.0;
    for (int i = 0; i < credits.length; i++) {
        weightedSum += marks[i] * credits[i];
        totalCredits += credits[i];
    }
}

```

```

return weightedSum / totalCredits
}

```

```

public static void main( String[] args) {
    Student student = new Student();
    student.acceptDetails();
    student.displayDetails();
    double sgpa = student.calculate SGPA();
    System.out.println("SGPA: " + sgpa);
}
}

```

Output → Enter the number of students: 1
 Enter USN: 1BM23CS299
 Enter Name: Sanchit Mehta
 Enter number of subjects: 3
 Enter the credits for subject 1: 4
 Enter the ^{Marks} credits for subject 1: 89
 Enter the credits for subject 2: 3
 Enter the marks for subject 2: 94
 Enter the credits for subject 3: 1
 Enter the marks for subject 3: 97

USN : IBM23CS299

Name: Sanchit Mehta

Credits and marks:

Subject 1 : credits = 4 , marks = 89

Subject 2 : credits = 3 , marks = 94

Subject 3 : credits = 1 , marks = 97

SGPA : 9.8

N
3/10/24

Code:

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

```
class Student {
```

```
    String usn;
```

```
    String name;
```

```
    int[] credits;
```

```
    int[] marks;
```

```
    public Student(int numSubjects) {
```

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

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

```
    }
```

```
    public void acceptDetails() {
```

```
        Scanner scanner = new Scanner(System.in);
```

```
        System.out.print("Enter USN: ");
```

```
        usn = scanner.nextLine();
```

```
        System.out.print("Enter Name: ");
```

```
        name = scanner.nextLine();
```

```
        for (int i = 0; i < credits.length; i++) {
```

```
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
```

```
            credits[i] = scanner.nextInt();
```

```
            System.out.print("Enter marks for subject " + (i + 1) + ": ");
```

```
            marks[i] = scanner.nextInt();
```

```
        }
```

```

    }
    public void displayDetails() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        for (int i = 0; i < credits.length; i++) {
            System.out.println("Subject " + (i + 1) + " - Credits: " + credits[i] + ", Marks: "
+ marks[i]);
        }
    }
    public double calculateSGPA() {
        double totalCredits = 0;
        double totalGradePoints = 0;
        for (int i = 0; i < credits.length; i++) {
            double gradePoint = getGradePoint(marks[i]);
            totalGradePoints += gradePoint * credits[i];
            totalCredits += credits[i];
        }
        return totalCredits == 0 ? 0 : totalGradePoints / totalCredits;
    }
    public double getGradePoint(int mark) {
        if (mark >= 90) return 10.0;
        else if (mark >= 80) return 9.0;
        else if (mark >= 70) return 8.0;
        else if (mark >= 60) return 7.0;
        else if (mark >= 50) return 6.0;
        else if (mark >= 40) return 5.0;
        else return 0.0; // Fail
    }
}

public class lab2 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter number of subjects: ");
        int numSubjects = scanner.nextInt();
        Student student = new Student(numSubjects);

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

        double sgpa = student.calculateSGPA();
        System.out.printf("SGPA: %.2f\n", sgpa);
    }
}

```

OUTPUT

```
Microsoft Windows [Version 10.0.22631.4060]
(c) Microsoft Corporation. All rights reserved.

C:\Users\sanch>cd C:\Users\sanch\Desktop\java
C:\Users\sanch\Desktop\java>javac lab2.java
C:\Users\sanch\Desktop\java>java lab2
Enter number of subjects: 5
Enter USN: 1BM23CS299
Enter Name: Sanchit
Enter credits for subject 1: 4
Enter marks for subject 1: 99
Enter credits for subject 2: 3
Enter marks for subject 2: 89
Enter credits for subject 3: 3
Enter marks for subject 3: 85
Enter credits for subject 4: 2
Enter marks for subject 4: 90
Enter credits for subject 5: 1
Enter marks for subject 5: 87
USN: 1BM23CS299
Name: Sanchit
Subject 1 - Credits: 4, Marks: 99
Subject 2 - Credits: 3, Marks: 89
Subject 3 - Credits: 3, Marks: 85
Subject 4 - Credits: 2, Marks: 90
Subject 5 - Credits: 1, Marks: 87
SGPA: 9.46
```

```
C:\Users\sanch\Desktop\java>java lab2
Enter number of subjects: 2
Enter USN: 1bm23cs123
Enter Name: xyz
Enter credits for subject 1: 1
Enter marks for subject 1: 78
Enter credits for subject 2: 5
Enter marks for subject 2: 98
USN: 1bm23cs123
Name: xyz
Subject 1 - Credits: 1, Marks: 78
Subject 2 - Credits: 5, Marks: 98
SGPA: 9.67
```

```
C:\Users\sanch\Desktop\java>java lab2
Enter number of subjects: 4
Enter USN: 1bm23cs189
Enter Name: piyush
Enter credits for subject 1: 4
Enter marks for subject 1: 80
Enter credits for subject 2: 3
Enter marks for subject 2: 87
Enter credits for subject 3: 2
Enter marks for subject 3: 92
Enter credits for subject 4: 1
Enter marks for subject 4: 67
USN: 1bm23cs189
Name: piyush
Subject 1 - Credits: 4, Marks: 80
Subject 2 - Credits: 3, Marks: 87
Subject 3 - Credits: 2, Marks: 92
Subject 4 - Credits: 1, Marks: 67
SGPA: 9.00
```


LABORATORY PROGRAM – 3

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

19/10/2024

VISION
Date _____ Page No. _____

Q Create a class Book which contains 4 members :
name, author, price, num_pages. Include a
constructor to set values for the members
Include the members to set and get the
details of the objects. Include a to
String() method that would display the
complete details of the book. Develop a
Java program to create n book objects.

Ans

```
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; per 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 set NumPages (int numPages) {  
    this.numPages = numPages;  
}
```

```
public String getName() {  
    return name;  
}
```

```
public String get Author() {  
    return Author;  
}
```

```
public String getPrice() {  
    return price;  
}
```

```
public int getNumPages() {  
    return numPages;  
}
```

```
public String toString() {
```

```
    return "Book Name:" + name + "\n" + "Author:"  
    + Author + "\n" + "Price:" + price + "\n" + "Number of  
    Pages:" + numPages;  
}
```

```
public class Main2 {
```

```
    public static void main( String[] args) {
```

```
        try( Scanner scanner = new Scanner(System.in))
```

```
        { System.out.println("Enter the no. of books:");
```

```
            int n = scanner.nextInt();
```

```
            scanner.nextLine();
```



```

Book[] books = new Book[n];

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

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

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

    System.out.print("No. of Pages: ");
    int numPages = Scanner.nextInt();

    Scanner.nextLine();

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

```

```

System.out.println("\nBook Details:");
for (Book book : books) {
    System.out.println(book);
    System.out.println();
}
}
}
}

```


Output: → Enter number of books: 3.

Enter details for book1:

Name: The Alchemist

Author: Sanchit

Price: 7999.99

No. of Pages: 289

Enter details for book2:

Name: Three men in a Boat

Author: Pyrusli

Price: 101.99

No. of Pages: 34

Enter details for book3:

Name: xyz

Author: abc

Price: 171.99

No. of Pages: 91

Code:

```
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;
    }
}
```

```

    }

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

public class LAB3{

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

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

        Book[] books = new Book[n];

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

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

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

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

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

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

        System.out.println("\nBook Details:");
        for (Book book : books) {
            System.out.println(book);
            System.out.println();
        }

        scanner.close();
    }
}

```

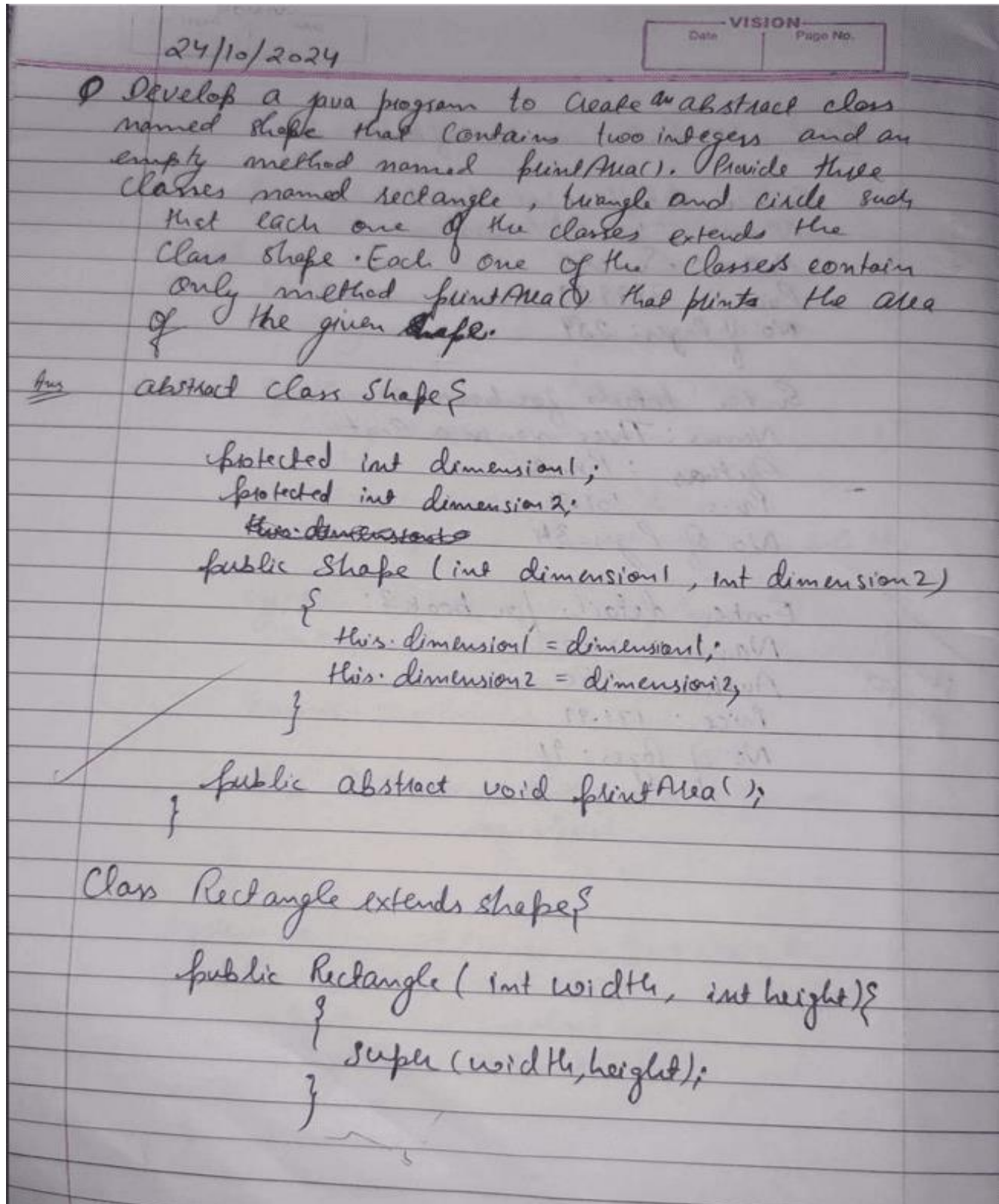
```
}  
}
```

OUTPUT

```
C:\Users\sanch\Desktop\java>java LAB3  
Enter the number of books: 3  
  
Enter details for book 1:  
Name: The Alchemist  
Author: Sanchit  
Price: 7999.99  
Number of Pages: 289  
  
Enter details for book 2:  
Name: Three men in a Boat  
Author: Piyusg  
Price: 101.99  
Number of Pages: 34  
  
Enter details for book 3:  
Name: xyz  
Author: abc  
Price: 171.99  
Number of Pages: 91  
  
Book Details:  
Book Name: The Alchemist  
Author: Sanchit  
Price: $7999.99  
Number of Pages: 289  
  
Book Name: Three men in a Boat  
Author: Piyusg  
Price: $101.99  
Number of Pages: 34  
  
Book Name: xyz  
Author: abc  
Price: $171.99  
Number of Pages: 91
```

LABORATORY PROGRAM – 4

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




```

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

```

Class Triangle extends Shape {

public Triangle extends Shape {

public Triangle (int base , int height) {

super (base, height);

public void printArea() {

double area = 0.5 * dimension1 * dimension2;

System.out.println("Area of Triangle: " + area);

}

Class Circle extends Shape {

public Circle (int radius) {

super (radius, 0);

}

public void printArea () {

double area = Math.PI * dimension1 * dimension2;

System.out.println("Area of Circle: " + area);

```
pre public class lab4 {
```

```
    public static void main(String[] args)
    {
```

```
        Rectangle
        Shape s = new Rectangle(5, 10);
```

```
Rectangle.printArea();
```

```
        rectangle.printArea();
```

```
        Shape Triangle = new Triangle(5, 10);
```

```
        triangle.printArea();
```

```
        Shape circle = new Circle(7);
```

```
        circle.printArea();
```

```
    }
```

```
}
```

Output:- Area of Rectangle : 50

Area of Triangle : 25.0

Area of Circle : 153.93804002189

24/10/24

Code:

```
abstract class Shape {
    int dimension1;
    int dimension2;
    Shape(int dimension1, int dimension2) {
        this.dimension1 = dimension1;
        this.dimension2 = dimension2;
    }
    public abstract void printArea();
}
class Rectangle extends Shape {
    public Rectangle(int width, int height) {
```



```

        super(width, height);
    }

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

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

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

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

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

public class lab4 {
    public static void main(String[] args) {
        Shape rectangle = new Rectangle(5, 10);
        rectangle.printArea();

        Shape triangle = new Triangle(5, 10);
        triangle.printArea();

        Shape circle = new Circle(7);
        circle.printArea();
    }
}

```

OUTPUT

```

Microsoft Windows [Version 10.0.22631.4460]
(c) Microsoft Corporation. All rights reserved.

C:\Users\sanch>cd C:\Users\sanch\Desktop\java
C:\Users\sanch\Desktop\java>javac lab4.java
C:\Users\sanch\Desktop\java>java lab4
Area of Rectangle: 50
Area of Triangle: 25.0
Area of Circle: 153.93804002589985

```

LABORATORY PROGRAM – 5

Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Create a class Account that stores customer name, account number and type of account. From this derive the classes Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:

- Accept deposit from customer and update the balance.
- Display the balance.
- Compute and deposit interest
- Permit withdrawal and update the balance

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

7/1/24

VISION Page No.

Q Develop a java program to create a class Bank that maintains two kinds of accounts 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 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 task:

- Accept deposit from customer and update the balance
- Display the balance
- Compute the deposit interest
- Permit withdrawal and update the balance

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

class

```
import java.util.Scanner
```

```
Class Account {
```

```
    String CustomerName;
```

```
    String typeAccount;
```

```
    double balance;
```

```
    long double accnum;
```

```
    Account(String CustomerName, String typeAccount,  
            double balance, double long accnum)
```

```
    {
```

```
        this.CustomerName = CustomerName;
```

```
        this.typeAccount = typeAccount;
```

```
        this.balance = balance;
```

```
        this.accnum = accnum;
```

```
    }
```

```
    public void deposit(double amount)
```

```
    {
```

```
        if (amount > 0)
```

```
        {
```

```
            if balance += amount;
```

```
        }
```

```
        else
```

```
        {
```

```
            System.out.println("Invalid Amount");
```

```
        }
```

```
    }
```



```

public void display()
{
    System.out.println("The balance is " + balance);
}

```

Class ~~Sav~~ ~~acct~~ extends Account {

double interestRate;

Savacc(String CustomerName, double balance, long
accnum, double interestRate)

```

{
    Super( CustomerName, balance, accnum,
            interestRate);
    this.interestRate = interestRate;
}

```

public CompoundInterest()

```

{
    double interest = balance * (interestRate/100);
    deposit(interest);
}

```

```

System.out.println("Interest Compound=" +
                    interest);
}

```

public withdraw(double amount)

```

{
    if (amount > 0 & & amount <= balance)
    {
        balance -= amount;
    }
}

```

```

else
{
    System.out.println("Invalid amount");
}
}

```

class curacc extends Account

```

{
    final double MinBal = 4000;
    final double Service_Charge = 100;

    public void withdraw (double amount)
    {
        if (amount > 0 && amount <= balance)
        {
            balance -= amount;
            System.out.println("withdraw" +
                               amount);
            checkMinBalance();
        }
    }
}

```

```

public void checkMinBalance()
{
    else
    {
        System.out.println("Invalid
                             amount");
    }
}

```

public void checkMinBalance() {

if (balance < Min-Bal)

{

balance -= Service-Charge;

System.out.println("Minimum Balance is not maintained");

}

}

public class Bank {

public static void main(String[] args) {

SavAcc savings = new SavAcc("Anam", 123457911L,
5500, 9.4);

savings.display();

savings.deposit(1000);

savings.display();

savings.withdraw(500);

savings.display();

savings.compoundInterest();

savings.display();

System.out.println("Current account operation

Cur Acc Current = new CurAcc ("sanchit",
1234567899C, 2000);
current.display();
current.deposit(500);
current.display();
current.withdraw(1200);
current.display();

Output → The balance is: 5500
Deposited: 1000
The balance is: 6500
Withdrawn: 500
The balance is: 6000
Deposited: 564
~~The~~ Interest compounded: 564
The balance is: 6564

Current account operations

The balance is: 2000
Deposited: 500
The balance is: 2500
Withdrawn: 1200
The balance is: 1300

✓ 28/11/24
LS

Code:

```
import java.util.Scanner;

class Account {
    String customerName;
    long accountNum;
    double balance;

    public Account(String customerName, long accountNum, double balance) {
        this.customerName = customerName;
        this.accountNum = accountNum;
        this.balance = balance;
    }

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

    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("Withdrew: " + amount);
        } else {
            System.out.println("Invalid request or insufficient funds");
        }
    }

    public void display() {
        System.out.println("The balance is: " + balance);
    }
}

class SavAcc extends Account {
    double interestRate;

    public SavAcc(String customerName, long accountNum, double balance, double interestRate) {
        super(customerName, accountNum, balance);
        this.interestRate = interestRate;
    }

    public void compoundInterest() {
        double interest = balance * (interestRate / 100);
        deposit(interest);
        System.out.println("Interest compounded: " + interest);
    }
}
```

```

    }
}

class CurAcc extends Account {
    static final double minbal = 1000;
    static final double servicecharge = 100;

    public CurAcc(String customerName, long accountNum, double balance) {
        super(customerName, accountNum, balance);
    }

    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("Withdrew: " + amount);
            checkMinBalance();
        } else {
            System.out.println("Invalid request or insufficient balance");
        }
    }

    public void checkMinBalance() {
        if (balance < minbal) {
            balance -= servicecharge;
            System.out.println("Minimum balance not maintained. Service charge imposed: " +
servicecharge);
        }
    }
}

public class lab5 {
    public static void main(String[] args)
    {

        SavAcc savings = new SavAcc("aman", 123457911, 5500.0, 9.4);
        savings.display();
        savings.deposit(1000.0);
        savings.display();
        savings.withdraw(500.0);
        savings.display();
        savings.compoundInterest();
        savings.display();

        System.out.println("current account operations\n");
        CurAcc current = new CurAcc("sanchit", 1234567899, 2000.0);
        current.display();
        current.deposit(500.0);
        current.display();
    }
}

```

```
        current.withdraw(1200.0);  
        current.display();  
    }  
}
```

OUTPUT

```
C:\1bm23cs299>java lab5  
Enter customer name for savings account: sanchit  
Enter account number for savings account: 111  
Enter initial balance for savings account:  
23  
Enter interest rate for savings account: 12  
The balance is: 23.0  
Enter the amount to deposit in savings account: 222  
Deposited: 222.0  
The balance is: 245.0  
Enter the amount to withdraw from savings account: 21  
Withdrew: 21.0  
The balance is: 224.0  
Deposited: 26.88  
Interest compounded: 26.88  
The balance is: 250.88  
  
Enter customer name for current account: Enter account number for current account: xyz  
Enter initial balance for current account: 1111  
The balance is: 1111.0  
  
Enter the amount to deposit in current account: 125678  
Deposited: 125678.0  
The balance is: 126789.0  
Enter the amount to withdraw from current account: 2223  
Withdrew: 2223.0  
The balance is: 124566.0
```

LABORATORY PROGRAM – 6

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

Q 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 5 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 5 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.

```
package cie;
public class Internals {
    int[] marks = new int[5];

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

    public void display() {
        System.out.println("CIE Marks:\n");
        for (int num : marks) {
            System.out.println(num);
        }
    }

    public int getTotalCIEMarks() {
        int total = 0;
        for (int num : marks) {
            total += num;
        }
        return total;
    }
}
```

```
package see;
```

```
import cse.student;
```

```
public class External extends Student {
```

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

```
    public External(String USN, String name,  
        int sem, int[] seemarks)
```

```
    {
```

```
        super(USN, name, sem);
```

```
        this.seemarks = seemarks;
```

```
    }
```

```
    public void display() {
```

```
        System.out.println("SEE marks are :\n");
```

```
        for (int num : seemarks)
```

```
        {
```

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

```
        }
```

```
public int public int gethalf() {
```

```
    int total = 0;
```

```
    for (int num : seemarks)
```

```
    {
```

```
        total += num;
```

```
    }
```

```
    return total / 2;
```

```
    }
```

```
}
```


package c1;

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 ("Student Details:");

System.out.println ("USN:" + USN);

System.out.println ("Name:" + name);

System.out.println ("Semester:" + sem);

}

```
import java.util.Scanner;
import cse.Student;
import cse.Intervals;
import cse.External;
```

```
public class Lab6 {
```

```
    public static void main(String[] args)
```

```
    {
        Scanner s = new Scanner(System.in);
```

```
        System.out.println("Enter number of  
students:");
```

```
        int n = s.nextInt();
```

```
        s.nextLine();
```

```
        for (int i = 1; i <= n; i++)
```

```
        {
            System.out.println("Enter details  
of student " + i + ":");
```

```
            System.out.println("Enter USN:");
            String USN = s.nextLine();
```

```
            System.out.println("Enter name:");
            String name = s.nextLine();
            s.nextLine();
```

```
            Student stud = new Student(USN,  
                                         name, sem);
```

```
            stud.display();
```



```
System.out.println("Enter CIE marks for 5  
subjects:");
```

```
int[] smarks = new int[5];  
for (int j = 0; j < 5; j++)  
{  
    smarks[j] = s.nextInt();  
}
```

```
External externalmarks = new External(  
    USN, name, sem, smarks);
```

```
externalmarks.display();
```

```
    s.nextLine();  
}
```

```
    s.close();  
}
```

```
}
```

Output :- Enter number of students.
2

Enter details of Student 1:

Enter USN:

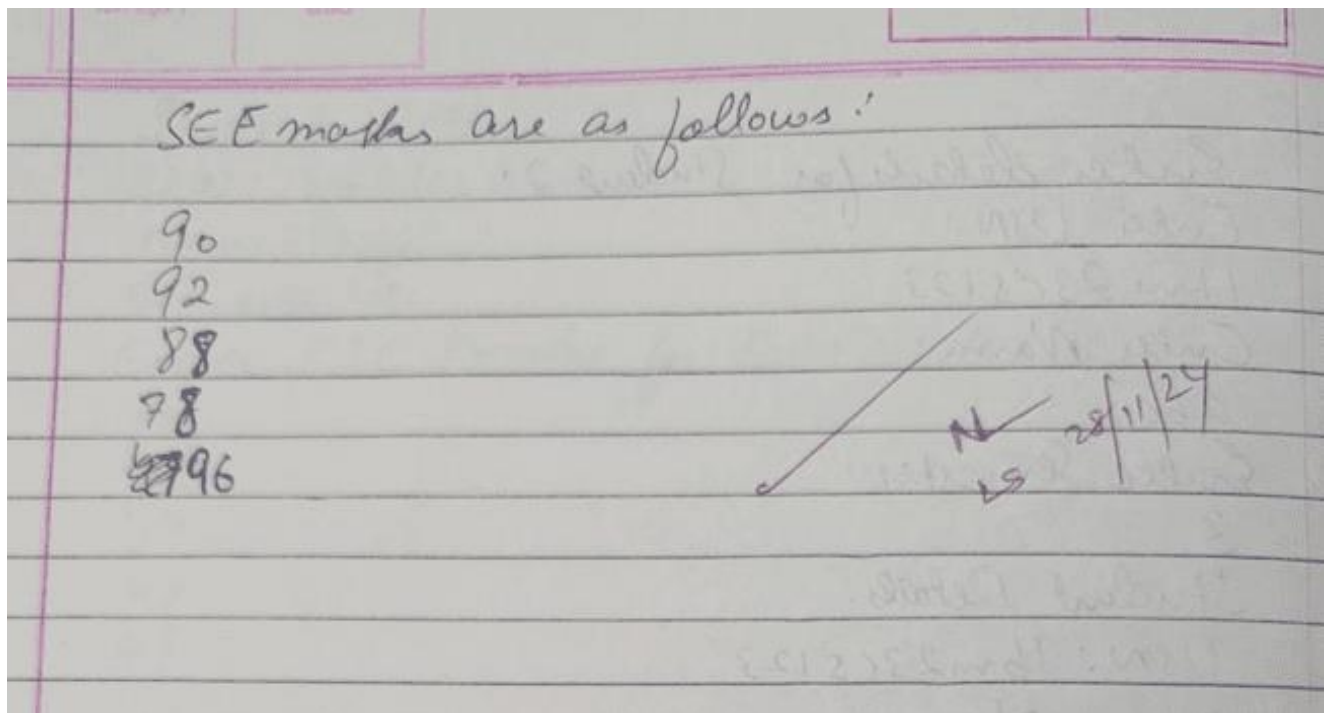
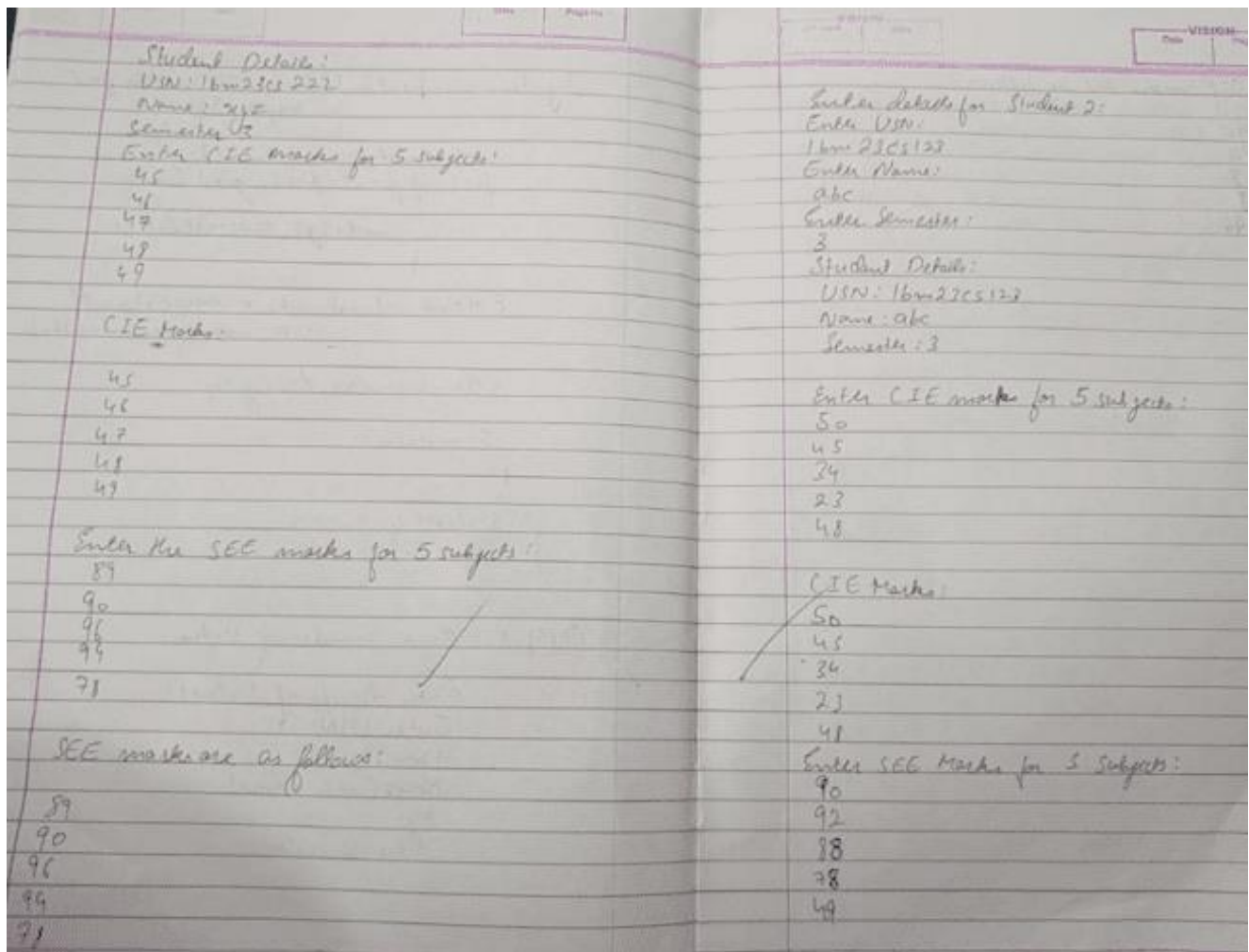
1bm23cs222

~~Enter~~ Enter Name:

xyz

Enter semester:

3



Code:

```
package cie;
public class Internals {
    int[] marks = new int[5];

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

    public void display() {
        System.out.println("CIE Marks: \n");
        for (int num : marks) {
            System.out.println(num);
        }
    }
}

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("Student Details:");
        System.out.println("USN: " + USN);
        System.out.println("Name: " + name);
        System.out.println("Semester: " + sem);
    }
}

package see;
import cie.Student;

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

    public External(String USN, String name, int sem, int[] seemarks) {
        super(USN, name, sem);
        this.seemarks = seemarks;
    }

    public void display() {
        System.out.println("SEE Marks are: \n");
        for (int num : seemarks) {
            System.out.println(num);
        }
    }
}

import java.util.Scanner;
import cie.Student;
1
```

```

import cie.Internals;
import see.External;

public class lab6 {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);

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

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

            System.out.println("Enter USN: ");
            String USN = s.nextLine();

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

            System.out.println("Enter Semester: ");
            int sem = s.nextInt();
            s.nextLine();

            Student stud = new Student(USN, name, sem);
            stud.display();

            int[] marks = readMarks(s, "CIE");
            Internals internalMarks = new Internals(marks);
            internalMarks.display();

            int[] smarks = readMarks(s, "SEE");
            External externalMarks = new External(USN, name, sem, smarks);
            externalMarks.display();

            int[] finalMarks = new int[5];
            System.out.println("Final Marks (CIE + 1/2 SEE) for each subject:");
            for (int j = 0; j < 5; j++) {
                finalMarks[j] = marks[j] + (smarks[j] / 2);
                System.out.println("Subject " + (j + 1) + ": " + finalMarks[j]);
            }
        }

        s.close();
    }

    public static int[] readMarks(Scanner s, String examType) {
        System.out.println("Enter the " + examType + " marks (5 subjects): ");
    }

```



```
int[] marks = new int[5];
for (int i = 0; i < 5; i++) {
    marks[i] = s.nextInt();
}
return marks;
}
}
```

OUTPUT

```
Enter the number of students: 2
Enter details for student 1:
USN: 1bm23
Name: sam
Semester: 3
Enter internal marks for 5 courses:
30 39 29 28 30
Enter SEE marks for 5 courses:
86 87 89 90 99
Enter details for student 2:
USN: 1bm24
Name: xyz
Semester: 2
Enter internal marks for 5 courses:
39 34 35 36 34
Enter SEE marks for 5 courses:
99 98 87 86 80

Final Marks of Students:
USN: 1bm23
Name: sam
Semester: 3
Internal Marks: 30 39 29 28 30
SEE Marks: 86 87 89 90 99
Final Marks: 116 126 118 118 129

USN: 1bm24
Name: xyz
Semester: 2
Internal Marks: 39 34 35 36 34
SEE Marks: 99 98 87 86 80
Final Marks: 138 132 122 122 114
```

LABORATORY PROGRAM - 7

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

21/11/24

Q Write a Program that demonstrates handling of exceptions in inheritance tree. Create a base class called Father and a 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 the 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.

Ans Class WrongAgeException extends Exception {

public String toString() {

return "WrongAgeException";

}

Class Father {

int age;

Father(int age) throws WrongAgeException {
if (age < 0)

throw new WrongAgeException(

this.age = age;

}

```

        int getAge();
        return age;
    }

    class Son extends Father {
        int sonAge;
        Son(int fatherAge, int sonAge) throws WrongAgeException {
            super(fatherAge);
            if (sonAge < 0 || sonAge > fatherAge) {
                throw new WrongAgeException();
            }
            this.sonAge = sonAge;
        }

        int getSonAge() {
            return sonAge;
        }
    }

    public class ExceptionHandlingInheritance {
        public static void main(String[] args) {
            try {
                Father f = new Father(12);
                Son s = new Son(f.getAge(), -23);
            } catch (WrongAgeException e) {
                System.out.println("Exception Caught: " + e);
            }
        }
    }

```

Output

```

Enter father's age: 12
Father's age: 12
Enter son's age: -23
Exception Caught: WrongAgeException

```

```

        System.out.println("Father's age: " + f.getAge());
        Son s = new Son(f.getAge(), -23);
        System.out.println("Son's age: " + s.getSonAge());
    }
}

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

Output
Enter father's age: 12
Father's age: 12
Enter son's age: -23
Exception Caught: WrongAgeException

```

VISION
 Date _____ Page No. _____

Enter father's age: -33
 Exception Caught: WrongAgeException

28/11/24

Code:

```
import java.util.Scanner;

class WrongAgeException extends Exception {
    public String toString() {
        return "WrongAgeException";
    }
}

class Father {
    int age;

    Father(int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException();
        }
        this.age = age;
    }

    int getAge() {
        return age;
    }
}

class Son extends Father {
    int sonAge;

    Son(int fatherAge, int sonAge) throws WrongAgeException {
        super(fatherAge);
        if (sonAge < 0 || sonAge >= fatherAge) {
            throw new WrongAgeException();
        }
        this.sonAge = sonAge;
    }

    int getSonAge() {
        return sonAge;
    }
}

public class lab7 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        try {

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



```

    Father f1 = new Father(fatherAge);
    System.out.println("Father's age: " + f1.getAge());
    System.out.print("Enter Son's age: ");
    int sonAge = scanner.nextInt();

    Son s1 = new Son(fatherAge, sonAge);
    System.out.println("Son's age: " + s1.getSonAge());

    } catch (WrongAgeException e) {
        System.out.println("Exception caught: " + e);
    } finally {
        scanner.close();
    }
}
}

```

OUTPUT

```

C:\1bm23cs299>java lab7
Enter Father's age: 12
Father's age: 12
Enter Son's age: 34
Exception caught: WrongAgeException

```

```

C:\1bm23cs299>java lab7
Enter Father's age: 12
Father's age: 12
Enter Son's age: -23
Exception caught: WrongAgeException

```

```

C:\1bm23cs299>java lab7
Enter Father's age: -33
Exception caught: WrongAgeException

```

LABORATORY PROGRAM – 8

Write a program which creates two threads, one thread displaying “BMS College of Engineering” once every ten seconds and another displaying “CSE” once every two seconds

28/11/24

Q Write a program which creates two threads, one thread displaying “BMS College of Engineering” once every 10 secs and other displaying “CSE” every two seconds.

Class Thread extends Thread

String s;

int time;

Thread (String s , int time)

{

this.s = s;

this.time = time;

}

public void run() {

try {

while (true)

{

System.out.println(s);

Thread.sleep(time * 1000);

}

}

catch (Exception e) {

System.out.println(e);

}

}

}

```

public class Main {
    public static void main (String[] args)
    {
        threads t1 = new threads(
            "BMS College of Engineering",
            1);
        threads t2 = new threads("CSE", 2);

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

```

Output :

```

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

```

28/11/24

```

class threads extends Thread{
    String s; int time;
    threads(String s, int time){
        this.s = s;
        this.time = time;
    }
    public void run(){
        try{
            while(true){
                System.out.println(s);
                Thread.sleep(time * 1000);
            }
        }
        catch(Exception e){
            System.out.println(e);
        }
    }
}

public class lab8{
    public static void main(String[] args) {
        threads t1 = new threads("BMS College of Engineering", 10);
        threads t2 = new threads("CSE", 2);

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

```

OUTPUT

```

C:\lbm23cs299>java lab8
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE

```


LABORATORY PROGRAM - 9

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

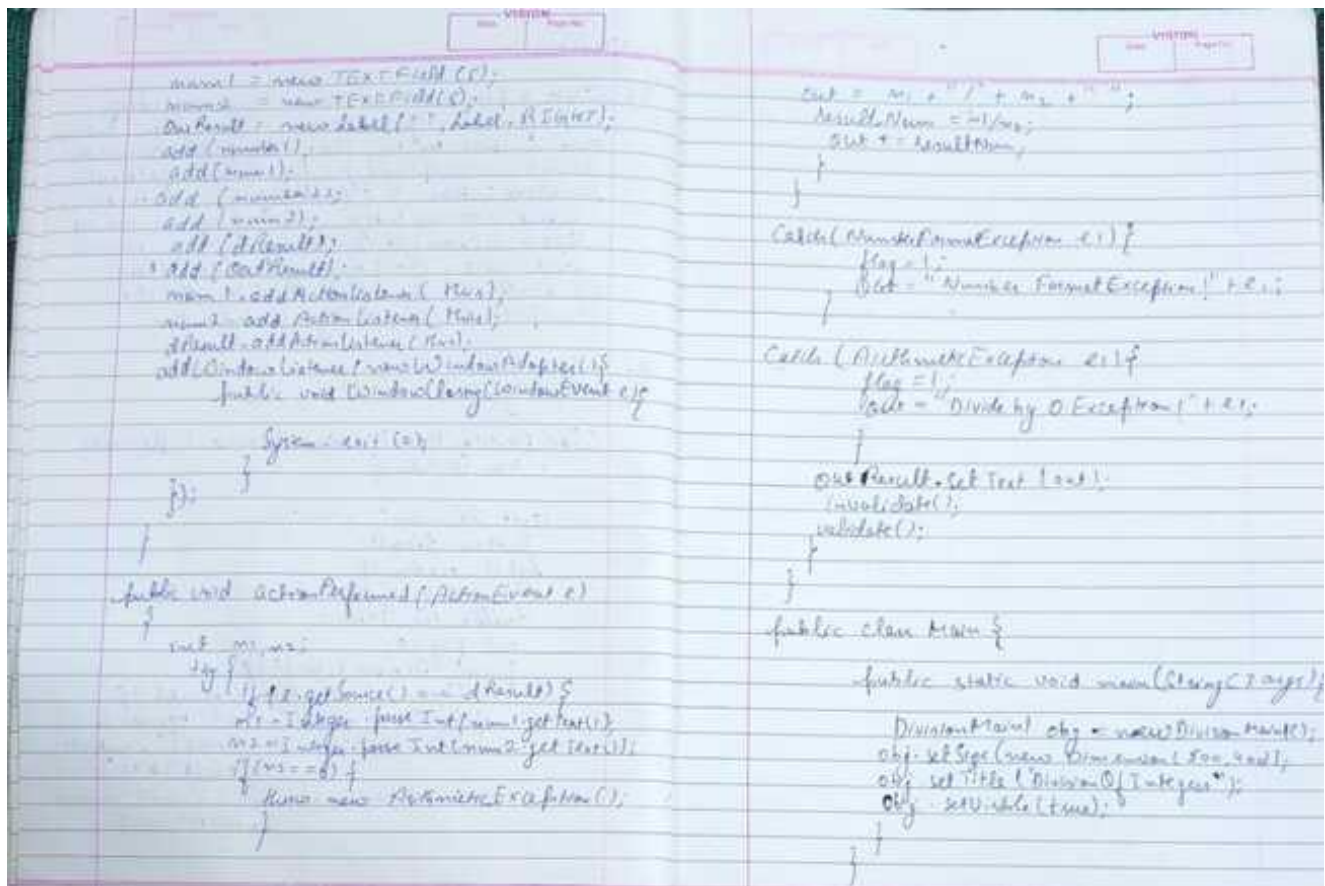
19/12/24

VISION
Date _____ Page 750

Q9 Write a program that creates a user interface to perform integer division. The user enters two numbers in the text fields, 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

Ans

```
import java.awt.*;  
import java.awt.event.*;  
  
class DivisionMain1 extends Frame implements  
    ActionListener {  
  
    TextField num1, num2;  
    Button dResult;  
    Label outResult;  
    String out = " ";  
    double resultNum;  
    int flag = 0;  
    public DivisionMain1() {  
        setLayout(new FlowLayout(1));  
        dResult = new Button("Result");  
        Label number1 = new Label("Number1:",  
            Label.RIGHT);  
        Label number2 = new Label("Number2:",  
            Label.RIGHT);
```



```

import java.awt.*;
import java.awt.event.*;
class DivisionMain1 extends Frame implements ActionListener
{
    TextField num1,num2;
    Button dResult;
    Label outResult;
    String out="";
    double resultNum;
    int flag=0;
    public DivisionMain1()
    {
        setLayout(new FlowLayout());
        dResult = new Button("Result:");
        Label number1 = new Label("Number 1:",Label.RIGHT);
        Label number2 = new Label("Number 2:",Label.RIGHT);
        num1=new TextField(5);
        num2=new TextField(5);
        outResult = new Label("",Label.RIGHT);
        add(number1);
        add(num1);
        add(number2);
        add(num2);
        add(dResult);
        add(outResult);
        num1.addActionListener(this);
        num2.addActionListener(this);
        dResult.addActionListener(this);
        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e)
            {

```

```

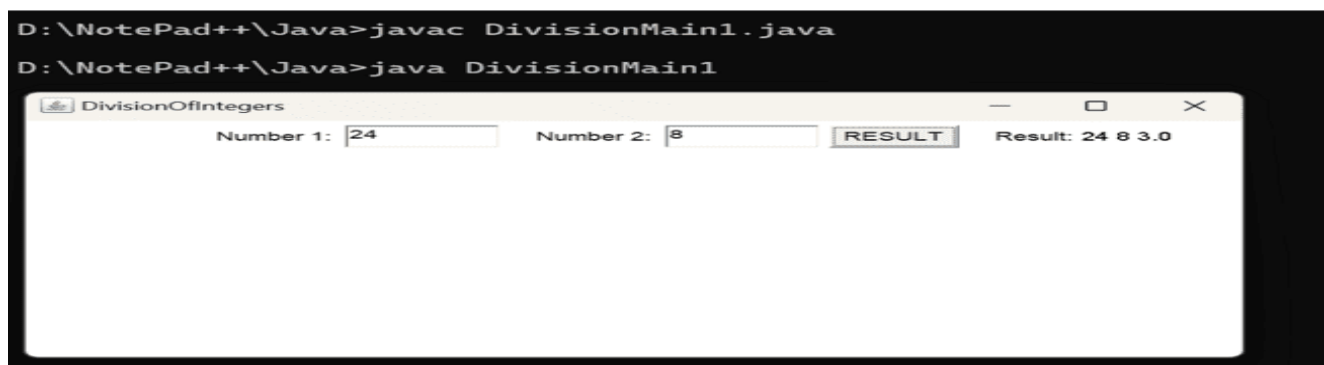
        System.exit(0);
    }
});
}
public void actionPerformed(ActionEvent e)
{
    int n1,n2;
    try
    {
        if (e.getSource() == dResult)
        {
            n1=Integer.parseInt(num1.getText());
            n2=Integer.parseInt(num2.getText());
            if(n2==0)
            {throw new ArithmeticException();}
            out=n1+"/"+n2+" ";
            resultNum=n1/n2;
            out+=resultNum;
        }
    }
    catch(NumberFormatException e1)
    {
        flag=1;
        out="Number Format Exception!" +e1;
    }
    catch(ArithmeticException e1)
    {
        flag=1;
        out="Divide by 0 Exception!" +e1;
    }
    outResult.setText(out);
    invalidate();
    validate();
}

}

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

```

OUTPUT



LABORATORY PROGRAM – 10

Demonstrate Inter process Communication and deadlock

19/11/24

Q1 Demonstrate Inter process Communication and deadlock.

Ans [11 Class 1]

```

int n;
boolean ValueSet = false;

Synchronized void get() {
    while (!ValueSet) {
        try {
            System.out.println("In Consumer Waiting");
            wait();
        } catch (InterruptedException e) {
            System.out.println("InterruptedException Caught");
        }
        System.out.println("Interrupted Exception Caught");
    }
    System.out.println("Get() = " + n);
    ValueSet = false;
    System.out.println("In Interrupted Producer");
    notify();
    return n;
}

```

```

Synchronized void put(int n) {
    while (ValueSet) {
        try {
            System.out.println("In Producer Waiting");
            wait();
        } catch (InterruptedException e) {
            System.out.println("InterruptedException Caught");
        }
        System.out.println("Interrupted Exception Caught");
    }
    ValueSet = true;
    System.out.println("Put() = " + n);
    notify();
}

```

Class Producer implements Runnable {

```

    private Queue q;
    public Producer(Queue q) {
        this.q = q;
        new Thread(this, "Producer").start();
    }
    public void run() {
        int i = 0;
        while (i < 10) {
            q.put(i);
            i++;
        }
    }
}

```

```

Class Consumer implements Runnable {
    private Queue q;
    public Consumer(Queue q) {
        this.q = q;
        new Thread(this, "Consumer").start();
    }
    public void run() {
        int i = 0;
        while (i < 10) {
            int n = q.get();
            System.out.println("Consumer = " + n);
            i++;
        }
    }
}

```

Class PCFTest {

```

    public static void main(String[] args) {
        Queue q = new Queue();
        new Producer(q);
        new Consumer(q);
        System.out.println("Enter Consumer - C to stop");
    }
}

```

Class A {

```

    Synchronized void foo(A a) {
        String name = Thread.currentThread().getName();
        System.out.println(name + " Entered A.foo");
        try {
            Thread.sleep(1000);
        } catch (Exception e) {
            System.out.println("A is Interrupted");
        }
        System.out.println(name + " Trying to call B.foo()");
        B b = new B();
        b.foo();
    }
    Synchronized void bar(A a) {
        System.out.println("A.foo()");
    }
    Synchronized void baz(A a) {
        System.out.println("A.baz()");
    }
}

```

Class B {

```

    Synchronized void foo(B b) {
        String name = Thread.currentThread().getName();
        System.out.println(name + " Entered B.foo");
        try {
            Thread.sleep(1000);
        } catch (Exception e) {
            System.out.println("B is Interrupted");
        }
        System.out.println(name + " Trying to call A.foo()");
        A a = new A();
        a.foo();
    }
    Synchronized void bar(B b) {
        System.out.println("B.bar()");
    }
    Synchronized void baz(B b) {
        System.out.println("B.baz()");
    }
}

```



```

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

```

Class Deadlock implements Runnable {

```

    A a = new A();

```

```

    B b = new B();

```

```

    Deadlock() {

```

```

        Thread.currentThread().setName("Main Thread");

```

```

        Thread t = new Thread(this, "Racing Thread");

```

```

        t.start(); a.foo(b);

```

```

        System.out.println("Back in main thread");

```

```

    }

```

```

    public void run() {

```

```

        b.bar(a);

```

```

        System.out.println("Back in other thread");

```

```

    }

```

```

    public static void main(String[] args) {

```

```

        new Deadlock();

```

```

    }

```

```

}

```

```

Code(i):
class Q {
int n;
boolean valueSet = false;
synchronized int get() {
while(!valueSet)
try {
System.out.println("\nConsumer waiting\n");
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
System.out.println("Got: " + n);
valueSet = false;
System.out.println("\nIntimate Producer\n");
notify();
return n;
}
synchronized void put(int n) {
while(valueSet)
try {
System.out.println("\nProducer waiting\n");
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
this.n = n;
valueSet = true;
System.out.println("Put: " + n);
System.out.println("\nIntimate Consumer\n");
notify();
}
}
class Producer implements Runnable {
Q q;
Producer(Q q) {
this.q = q;
new Thread(this, "Producer").start();
}
public void run() {
int i = 0;
while(i<15) {
q.put(i++);
}
}
}
class Consumer implements Runnable {
Q q;
Consumer(Q q) {
this.q = q;
new Thread(this, "Consumer").start();
}
public void run() {
int i=0;
while(i<15) {
int r=q.get();
System.out.println("consumed:"+r);
i++;
}
}
}
class PCFixed {

```

```

public static void main(String args[]) {
    Q q = new Q();
    new Producer(q);
    new Consumer(q);
    System.out.println("Press Control-C to stop.");
}
}

```

OUTPUT

```

Press Control-C to stop.
Put: 0

Intimate Consumer

Producer waiting
Got: 0
Intimate Producer
Put: 1
Intimate Consumer
consumed:0
Producer waiting
Got: 1
Intimate Producer
consumed:1
Put: 2
Intimate Consumer

Producer waiting
Got: 2
Intimate Producer
consumed:2
Put: 3
Intimate Consumer

```

Code(ii):

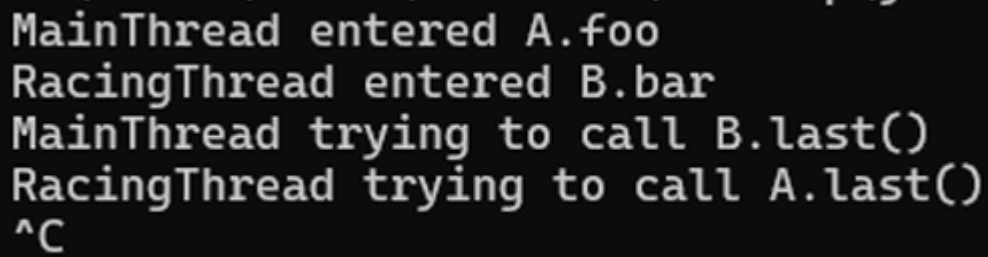
```

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

```

```
}  
class Deadlock implements Runnable  
{  
    A a = new A(); B b = new B();  
    Deadlock( ) {  
        Thread.currentThread().setName("MainThread");  
        Thread t = new Thread(this, "RacingThread");  
        t.start(); a.foo(b); // get lock on a in this thread.  
        System.out.println("Back in main thread");  
    }  
    public void run() { b.bar(a); // get lock on b in other thread.  
        System.out.println("Back in other thread");  
    }  
    public static void main(String args[]) { new Deadlock(); }  
}
```

OUTPUT



```
MainThread entered A.foo  
RacingThread entered B.bar  
MainThread trying to call B.last()  
RacingThread trying to call A.last()  
^C
```
