VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Bhuvan A (24BECS400)

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 **Bhuvan A(24BECS400),** 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.

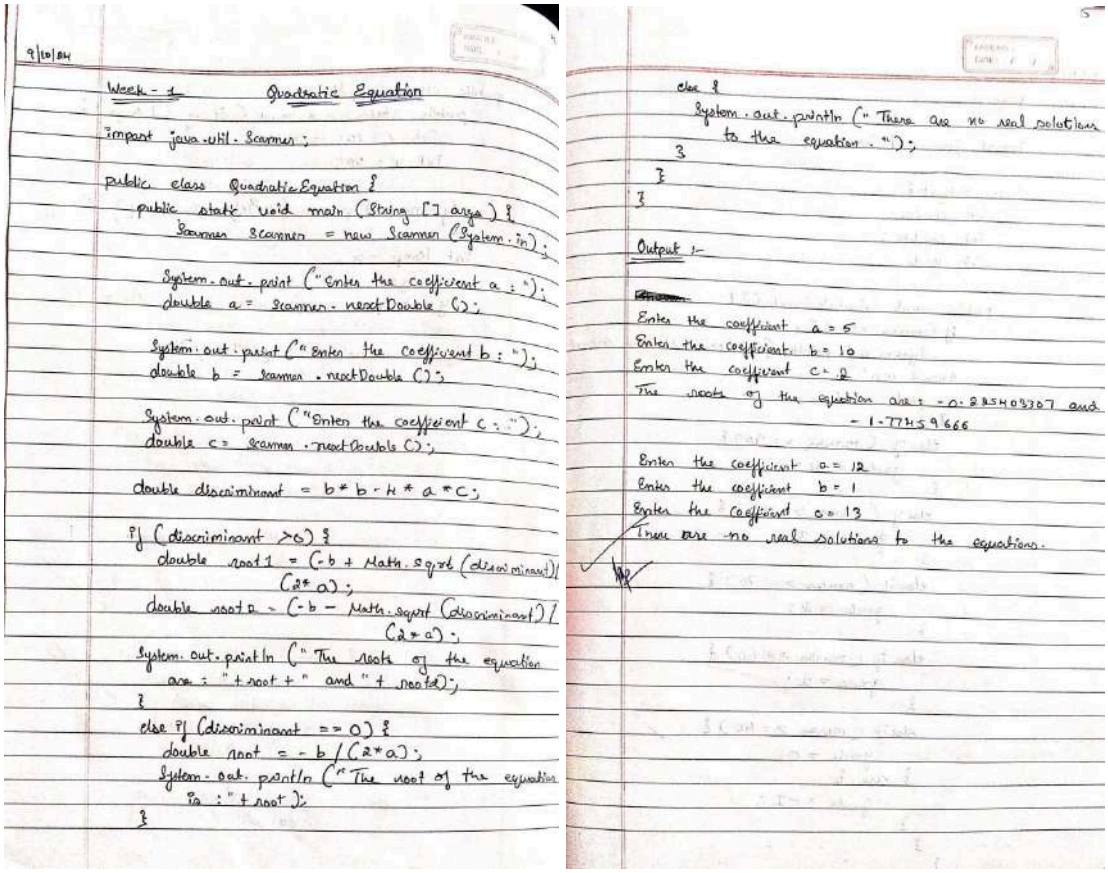
Dr. Prasad G R Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Github Link: https://github.com/abhuvan345/Java

Program 1

Develop a Java program that prints all real solutions to the quadratic equation ax2 +bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2 -4ac is negative, display a message stating that there are no real solutions.



9/10/24

Week - 1 Quadratic Equation

```
import java.util.Scanner;

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

        System.out.print ("Enter the coefficient a : ");
        double a = scanner.nextDouble ();

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

        System.out.print ("Enter the 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.println ("The roots of the equation
                are : " + root1 + " and " + root2);
        }
        else if (discriminant == 0) {
            double root = -b/(2*a);
            System.out.println ("The root of the equation
                is : " + root);
        }
        else {
            System.out.println (" There are no real solutions
                to the equation. ");
        }
    }
}
```

Output :-

```
Enter the coefficient a = 5
Enter the coefficient b = 10
Enter the coefficient c = 2
The roots of the equation are : -0.2254033307 and
    -1.7745966
Enter the coefficient a = 12
Enter the coefficient b = 1
Enter the coefficient c = 13
There are no real solutions to the equations.
```

Code:

```
import java.util.Scanner;

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

            System.out.println("Bhuvan. A");
            System.out.println("24BECS400");

        System.out.print("Enter the coefficient a: ");
        double a = scanner.nextDouble();
```

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

        System.out.print("Enter the 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.println("The roots of the equation are: " + root1 + " and " + root2);
        } else if (discriminant == 0) {
            double root = -b / (2 * a);
            System.out.println("The root of the equation is: " + root);
        } else {
            System.out.println("There are no real solutions to the equation.");
        }
    }
}
```

```
C:\Users\bmsce\Desktop>java QuadraticEquation
Bhuvan. A
24BECS400
Enter the coefficient a: 6
Enter the coefficient b: 17
Enter the coefficient c: 12
The roots of the equation are: -1.3333333333333333 and -1.5

C:\Users\bmsce\Desktop>
```

```
C:\Users\bmsce\Desktop>java QuadraticEquation
Enter the coefficient a: 10
Enter the coefficient b: 10
Enter the coefficient c: 10
There are no real solutions to the equation.
```

```
C:\Users\bmsce\Desktop>java QuadraticEquation
Enter the coefficient a: 5
Enter the coefficient b: 10
Enter the coefficient c: 2
The roots of the equation are: -0.2254033307585166 and -1.7745966692414832
```

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.

```
16/11/24
Week-2

import java.util.Scanner;

class Subject {
    int marks;
    Int credits;
    int grade;

    public void calculateGrade() {
        if (marks > 100) {
            System.out.println("Error, Marks cannot
            exceed 100");
            grade = -1
        }
        else if (marks >= 90) {
            grade = 10;
        }
        else if (marks >= 80) {
            grade = 9;
        }
        else if (marks >= 70) {
            grade = 8;
        }
        else if (marks >= 60) {
            grade = 7;
        }
        else if (marks >= 40) {
            grade = 0;
        } else {
            grade = -1;
        }
    }
}
```

```
class Student {
    String name;
    String usn;
    double SGPA;
    Scanner sc = new Scanner(System.in);
    Subject[] subjects;

    public Student(int numSubjects) {
        subjects = new Subject[numSubjects];
        for (int i = 0; i < numSubjects; i++) {
            subjects[i] = new Subject();
        }
    }

    public void getStudentDetails() {
        System.out.println("Enter name:");
        name = sc.nextLine();
        System.out.println("Enter USN:");
        Usn = sc.nextLine();
    }

    public void getMarks() {
        for (int i = 0; i < subjects.length; i++) {
            System.out.println("Enter marks for
                subjects" + (i+1) + ":");
            subjects[i].marks = sc.nextInt();
            System.out.println("Enter credits for
                subjects" + (i+1) + ":");
            subjects[i].credits = sc.nextInt();
            sc.nextLine();
            subjects[i].calculateGrade();
        }
    }
```

```
    public double computeSGPA() {
        double effectiveScore = 0.0;
        int totalCredits = 0;
        for (subject.grade == -1) {
            return -1.0;
        }
        effectiveScore += (subject.grade *
            subject.credits);
        totalCredits += subject.credits;
    }
    SGPA = (double) effectiveScore / (double)
        totalCredits;
    return SGPA;
  }
}

public class main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number of
            subjects:");
        int numSubjects = sc.nextInt();
        Student student = new Student(numSubjects);

        student.getStudentDetails();
        student.getMarks();

        double sgpa = student.computeSGPA();
        if (sgpa == -1.0) {
            System.out.println("Student Details:")
            System.out.println("Name:" + student.
                name);
```

Array of objects

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

Output :-

```
Enter your name : Bhuvan. A
Enter your USN : 2SD22CS100
Enter marks for Subject 1 : 90
Enter credits for Subject 1 : 4
Enter marks for Subject 2 : 98
Enter marks for Subject 3 : 96
Enter credit for Subject 3 : 4

Student Details :
Name : Bhuvan. A
USN : 2SD22CS100
SGPA : 10.0
```

Code:

```
import java.util.Scanner;

class Subject {

    int marks;
    int credits;
    int grade;

    public void calculateGrade() {
        if (marks > 100) {
            System.out.println("Error: Marks cannot exceed 100.");
            grade = -1;
        } else if (marks >= 90) {
            grade = 4; // A
        } else if (marks >= 80) {
            grade = 3; // B
        } else if (marks >= 70) {
            grade = 2; // C
        } else if (marks >= 60) {
            grade = 1; // D
        } else if (marks >= 40) {
            grade = 0; // E
        } else {
            grade = -1; // F
        }
    }
}

class Student {
    String name;
    String usn;
    double SGPA;

    Scanner sc = new Scanner(System.in);
    Subject[] subjects;

    public Student(int numSubjects) {
        subjects = new Subject[numSubjects];
        for (int i = 0; i < numSubjects; i++) {
            subjects[i] = new Subject();
        }
    }

    public void getStudentDetails() {
        System.out.println("Enter name: ");
        name = sc.nextLine();
```

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

    public void getMarks() {

        for (int i = 0; i < subjects.length; i++) {
            System.out.println("Enter marks for subject " + (i + 1) + ": ");
            subjects[i].marks = sc.nextInt();
            System.out.println("Enter credits for subject " + (i + 1) + ": ");
            subjects[i].credits = sc.nextInt();
            sc.nextLine();
            subjects[i].calculateGrade();
        }
    }

    public double computeSGPA() {
        double effectiveScore = 0.0;
        int totalCredits = 0;

        for (Subject subject : subjects) {
            if (subject.grade == -1) {
                return -1.0;
            }

            effectiveScore += (subject.grade * subject.credits);
            totalCredits += subject.credits;
        }

        SGPA = (double) effectiveScore / (double) totalCredits;
        return SGPA;
    }
}

public class Main {

    public static void main(String[] args) {
        System.out.println("Bhuvan. A\n" + "24BECS400");
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number of subjects: ");
        int numSubjects = sc.nextInt();
        Student student = new Student(numSubjects);
        student.getStudentDetails();
        student.getMarks();
        double sgpa = student.computeSGPA();

        if (sgpa == -1.0) {
```

```
            System.out.println("Error: Marks cannot exceed 100.");
        } else {
            System.out.println("Student Details:");
            System.out.println("Name: " + student.name);
            System.out.println("USN: " + student.usn);
            System.out.println("SGPA: " + sgpa);
        }
    }
}
```

```
C:\Users\bmsce\Desktop\24BECS400>java Main
Bhuvan. A
24BECS400
Enter the number of subjects:
3
Enter name:
Bhuvan. A
Enter USN:
24BECS400
Enter marks for subject 1:
90
Enter credits for subject 1:
4
Enter marks for subject 2:
98
Enter credits for subject 2:
3
Enter marks for subject 3:
96
Enter credits for subject 3:
4
Student Details:
Name: Bhuvan. A
USN: 24BECS400
SGPA: 4.0

C:\Users\bmsce\Desktop\24BECS400>java Main
Bhuvan. A
24BECS400
Enter the number of subjects:
3
Enter name:
Abhay
Enter USN:
24BECS404
Enter marks for subject 1:
90
Enter credits for subject 1:
4
Enter marks for subject 2:
89
Enter credits for subject 2:
4
Enter marks for subject 3:
85
Enter credits for subject 3:
3
Student Details:
Name: Abhay
USN: 24BECS404
SGPA: 3.3636363636363638
```

```
C:\Users\bmsce\Desktop\24BECS400>java Main
Bhuvan. A
24BECS400
Enter the number of subjects:
3
Enter name:
Sanketh
Enter USN:
24BECS422
Enter marks for subject 1:
99
Enter credits for subject 1:
4
Enter marks for subject 2:
98
Enter credits for subject 2:
4
Enter marks for subject 3:
89
Enter credits for subject 3:
3
Student Details:
Name: Sanketh
USN: 24BECS422
SGPA: 3.727272727272727
```

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.

```
8/10/24

Week - 3

import java.util.Scanner;

class Book {
    String name;
    String author;
    int price;
    int numpages;

    public Book (String name, String author, int price,
            int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }

    public String toString () {
        String bookDetails = "Book name : " + this.name
                + "\n" +
        "Author name : " + this.author + "\n" +
        "Price : " + this.Price + "\n" +
        "Number of Pages : " + this.numPages + "\n";
    return bookDetails;
    }
}
public class Main {
    public static void main (String args []) {
        Scanner scanner = new Scanner (System.in);
        System.out.println ("Enter the no. of books: ");
        int n = scanner.nextInt ();
```

```
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.next();
    System.out.println ("Author: ");
    String author = scanner.next ();
    System.out.println ("Price : ");
    int price = scanner.nextInt ();
    System.out.println ("Number of page : ");
    int numPages = scanner.nextInt ();

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

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

scanner.close ();
}
```

```
Output :-

Enter the number of books : 1

Enter details for books 1 :
Name : Java
Author : Seema Patil
Price : 450
Number of Pages : 650

Book Details :

Book name : Java
Author name : Seema Patil
Price : 450
Number of Pages : 650
```

Code:

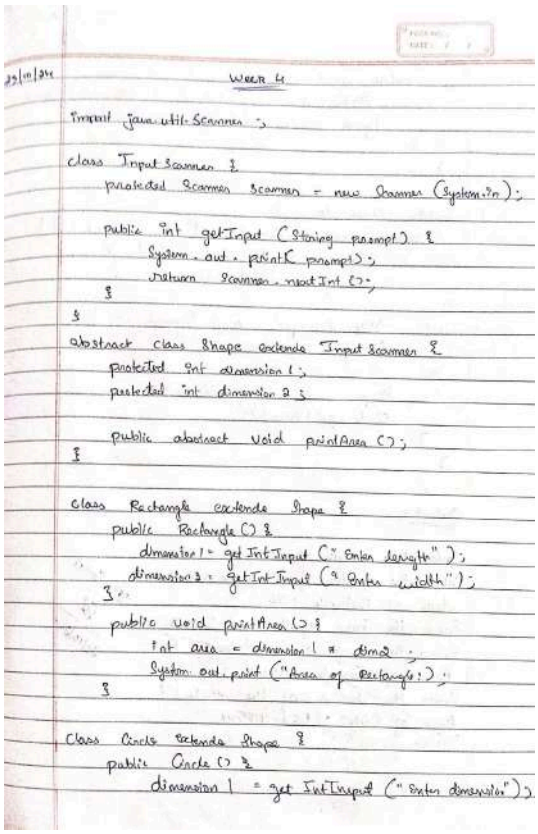
```
import java.util.Scanner;

// Define the Book class
class Book {
    String name;
    String author;
    int price;
    int numPages;

    // Parameterized constructor
    public Book(String name, String author, int price, int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }

    // toString method to return the book details

    public String toString() {
        String bookDetails = "Book name: " + this.name + "\n" +
                             "Author name: " + this.author + "\n" +
                             "Price: " + this.price + "\n" +
                             "Number of pages: " + this.numPages + "\n";
        return bookDetails;
    }
}

// Main class to run the program
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Read the number of books
            System.out.println("Bhuvan. A\n24BECS400");
        System.out.print("Enter the number of books: ");
        int n = scanner.nextInt();

        // Define an array of Book objects
        Book[] books = new Book[n];

        // Loop to read book details
        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for book " + (i + 1) + ":");
            System.out.print("Name: ");
            String name = scanner.next();
```

```
            System.out.print("Author: ");
            String author = scanner.next();
            System.out.print("Price: ");
            int price = scanner.nextInt();
            System.out.print("Number of pages: ");
            int numPages = scanner.nextInt();

            // Create a new Book object and add it to the array
            books[i] = new Book(name, author, price, numPages);
        }

        // Loop to display book details
        System.out.println("\nBook Details:");
        for (int i = 0; i < n; i++) {
            System.out.println(books[i].toString());
        }

        // Close the scanner
        scanner.close();
    }
}
```

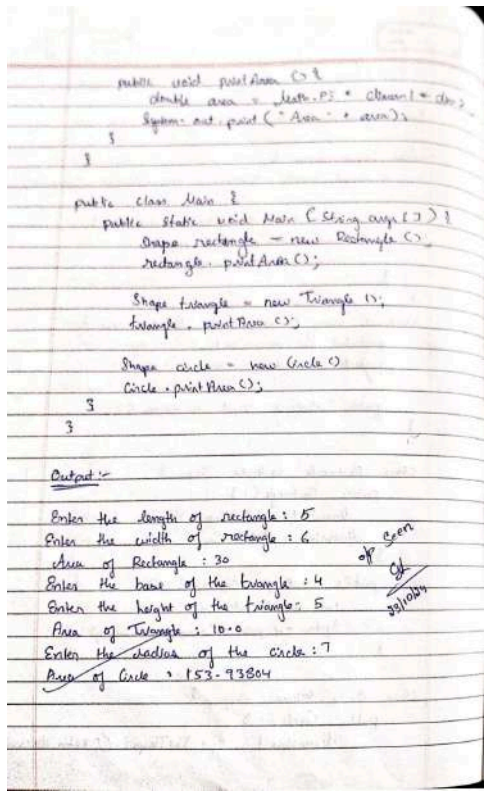
```
C:\Users\bmsce\Desktop\24BECS400>java Main1
Bhuvan.A
24BECS400
Enter the length of the rectangle: 5
Enter the width of the rectangle: 6
Area of Rectangle: 30
Enter the base of the triangle: 4
Enter the height of the triangle: 5
Area of Triangle: 10.0
Enter the radius of the circle: 7
Area of Circle: 153.93804002589985

C:\Users\bmsce\Desktop\24BECS400>
```

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.



```
25/10/24                WEEK 4

Import java.util.Scanner ;

class Input Scanner {
    protected Scanner scanner = new Scanner (System.in);

    public int getInput (String prompt) {
        System.out.print( prompt);
        return Scanner.nextInt ();
    }
}
abstract class Shape extends Input Scanner {
    protected int dimension 1;
    protected int dimension 2 ;

    public abstract void printArea ();
}

class Rectangle extends Shape {
    public Rectangle () {
        dimension 1 = get Int Input (" Enter length" );
        dimension 2 = get Int Input (" Enter width" );
    }
    public void printArea () {
        int area = dimension 1 * dim2 ;
        System out.print ("Area of Rectangle:) ;
    }

Class Circle extends Shape {
    public Circle () {
        dimension 1 = get Int Input (" Enter dimension" );
```



```
    public void printArea () {
        double area = Math.PI * dimen1 * dim2;
        System.out.print (" Area " + area);
    }
}

public class Main {
    public static void Main (String args[]) {
        Shape rectangle = new Rectangle ();
        rectangle.printArea ();

        Shape triangle = new Triangle ();
        triangle.printArea ();

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

Output :-

Enter the length of rectangle : 5
Enter the width of rectangle : 6
Area of Rectangle : 30
Enter the base of the triangle : 4
Enter the height of the triangle : 5
Area of Triangle : 10.0
Enter the radius of the circle : 7
Area of Circle : 153.93804
```

Code:

```
import java.util.Scanner;


class InputScanner {
    protected Scanner scanner = new Scanner(System.in);

    public int getIntInput(String prompt) {
        System.out.print(prompt);
        return scanner.nextInt();
    }
}


abstract class Shape extends InputScanner {
    protected int dimension1;
    protected int dimension2;
```

```
    public abstract void printArea();
}


class Rectangle extends Shape {
    public Rectangle() {
        dimension1 = getIntInput("Enter the length of the rectangle: ");
        dimension2 = getIntInput("Enter the width of the rectangle: ");
    }


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


class Triangle extends Shape {
    public Triangle() {
        dimension1 = getIntInput("Enter the base of the triangle: ");
        dimension2 = getIntInput("Enter the height of the triangle: ");
    }


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


class Circle extends Shape {
    public Circle() {
        dimension1 = getIntInput("Enter the radius of the circle: ");
    }


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


public class Main1 {
```

```
    public static void main(String[] args) {
        System.out.println("Bhuvan.A\n24BECS400");
        Shape rectangle = new Rectangle();
        rectangle.printArea();

        Shape triangle = new Triangle();
        triangle.printArea();

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

```
C:\Users\bmsce\Desktop\24BECS400>java Main
Bhuvan. A
24BECS400
Enter the number of books: 6
Enter details for book 1:
Name: Java
Author: SemmaPatil
Price: 450
Number of pages: 650
Enter details for book 2:
Name: DBMS
Author: Umadevi
Price: 600
Number of pages: 560
Enter details for book 3:
Name: COA
Author: Megha
Price: 890
Number of pages: 640
Enter details for book 4:
Name: LD
Author: Geetha
Price: 1000
Number of pages: 780
Enter details for book 5:
Name: DS
Author: Rajeshwari
Price: 800
Number of pages: 650
Enter details for book 6:
Name: Unix
Author: Ashvini
Price: 800
Number of pages: 430
```

```
Book Details:
Book name: Java
Author name: SemmaPatil
Price: 450
Number of pages: 650

Book name: DBMS
Author name: Umadevi
Price: 600
Number of pages: 560

Book name: COA
Author name: Megha
Price: 890
Number of pages: 640

Book name: LD
Author name: Geetha
Price: 1000
Number of pages: 780

Book name: DS
Author name: Rajeshwari
Price: 800
Number of pages: 650

Book name: Unix
Author name: Ashvini
Price: 800
Number of pages: 430
```

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.

12/10/24

Week 5

```
import java.util.Scanner;

class Account {
    private String custName;
    private String accNo;
    private double balance;

    public Account (String custName, String accNo,
        double balance) {
        this.custName = custName;
        this.accNo = accNo;
        this.balance = balance;
    }
    public double getBalance () {
        return this.Balance;
    }

    public void deposit (double amount) {
        if (amount >0) {
            this.balance + = amount;
            System.out.println ("The current ");
        }
        else {
            System.out.println ("Amount : ");
        }
    }

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

```
String accnt = sc.nextline ();
while (true) {
    System.out.print ("Enter your choice ");
    int choice = sc.nextInt ();

    Switch (choice) {
        case 1:
            System.out.println ("Enter balance");

        case 2:
            System.out.println ("Enter balance");

        default:
            System.out.println ("Invalid choice")
        }
    }
}
```

Output :

Saving account	Current account
Balance : 10000.00	Balance : 12005
Deposited : 100.00	Deposited : 12000.50
Balance : 11000.00	Balance : 1112500.00
Interest of 75.0 has been added	
Balance : 1575.0	
Withdraw : 300.0	
Balance : 1275.0	

Code:

```
import java.util.Scanner;

class Account {
    private String customerName;
    private String accountNumber;
    protected double balance;

    public Account(String customerName, String accountNumber) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.balance = 0.0;
    }

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

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

    public void withdraw(double amount) {
        if (amount <= balance) {
            balance -= amount;
            System.out.println("Withdraw amount: " + amount);
        } else {
            System.out.println("Insufficient balance for withdrawal!");
        }
    }

    protected double getBalance() {
        return balance;
    }
}

class SavAcct extends Account {
    private double interestRate;

    public SavAcct(String customerName, String accountNumber, double interestRate) {
        super(customerName, accountNumber);
        this.interestRate = interestRate;
    }

    public void computeAndDepositInterest() {
        double currentBalance = getBalance();
```

```
        double interest = currentBalance * interestRate / 100;
        deposit(interest);
        System.out.println("Interest deposited: " + interest);
    }
}

class CurAcct extends Account {
    private double minimumBalance;
    private double serviceCharge;

    public CurAcct(String customerName, String accountNumber, double minimumBalance, double
serviceCharge) {
        super(customerName, accountNumber);
        this.minimumBalance = minimumBalance;
        this.serviceCharge = serviceCharge;
    }

    public void withdraw(double amount) {
        if (getBalance() - amount < minimumBalance) {
            System.out.println("Service charge imposed: " + serviceCharge);
            deposit(-serviceCharge);
            System.out.println("Insufficient balance.");
        } else {
            super.withdraw(amount);
        }
    }
}

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


        System.out.println("Enter customer name for Savings Account:");
        String savingsCustomerName = scanner.nextLine();
        System.out.println("Enter account number for Savings Account:");
        String savingsAccountNumber = scanner.nextLine();
        System.out.println("Enter interest rate for Savings Account:");
        double interestRate = scanner.nextDouble();

        SavAcct savingsAccount = new SavAcct(savingsCustomerName, savingsAccountNumber,
interestRate);
        savingsAccount.deposit(1000);
        savingsAccount.computeAndDepositInterest();
        savingsAccount.displayBalance();
```

```
        System.out.println("Enter amount to withdraw from Savings Account:");
        double withdrawAmount = scanner.nextDouble();
        savingsAccount.withdraw(withdrawAmount);
        savingsAccount.displayBalance();


        scanner.nextLine();
        System.out.println("Enter customer name for Current Account:");
        String currentCustomerName = scanner.nextLine();
        System.out.println("Enter account number for Current Account:");
        String currentAccountNumber = scanner.nextLine();
        System.out.println("Enter minimum balance for Current Account:");
        double minimumBalance = scanner.nextDouble();
        System.out.println("Enter service charge for Current Account:");
        double serviceCharge = scanner.nextDouble();

        CurAcct currentAccount = new CurAcct(currentCustomerName, currentAccountNumber,
minimumBalance, serviceCharge);
        currentAccount.deposit(2000);
        currentAccount.displayBalance();

        System.out.println("Enter amount to withdraw from Current Account:");
        double currentWithdrawAmount = scanner.nextDouble();
        currentAccount.withdraw(currentWithdrawAmount);
        currentAccount.displayBalance();


        System.out.println("Enter amount to withdraw from Current Account (may incur service
charge):");
        currentWithdrawAmount = scanner.nextDouble();
        currentAccount.withdraw(currentWithdrawAmount);
        currentAccount.displayBalance();

        scanner.close();
    }
}
```

```
D:\24BECS400\week5>javac Bank.java

D:\24BECS400\week5>java Bank
Enter customer name for Savings Account:
Bhuvan. A
Enter account number for Savings Account:
20110215220
Enter interest rate for Savings Account:
2
Deposited amount: 1000.0
Deposited amount: 20.0
Interest deposited: 20.0
Balance amount: 1020.0
Enter amount to withdraw from Savings Account:
10
Withdraw amount: 10.0
Balance amount: 1010.0
Enter customer name for Current Account:
Sachin
Enter account number for Current Account:
2055425102
Enter minimum balance for Current Account:
1000000
Enter service charge for Current Account:
10
Deposited amount: 2000.0
Balance amount: 2000.0
Enter amount to withdraw from Current Account:
20000
Service charge imposed: 10.0
Deposited amount: -10.0
Insufficient balance.
Balance amount: 1990.0
Enter amount to withdraw from Current Account (may incur service charge):
10000
Service charge imposed: 10.0
Deposited amount: -10.0
Insufficient balance.
Balance amount: 1980.0

D:\24BECS400\week5>
```

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.

Week - 6

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

Main.java

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

public class Main {
    public class void void main (String args[]) {
        System.out.print("Enter no.of Students");
        int n = s.nextInt();
        External[] students = new External[n];

        for (int i =0 ; i <n ; i++) {
            System.out.println("Entering details for students
                + (i+1) + ":");
            students[i] = new External();
            System.out.println("In Final Marks of students
            for (int i = 0 ; i<n ; i++){
                System.out.println("Student " + (i+1) + ":");
                students[i].displayFinalMarks();
            }
        }
    }
```

Internals.java

```
package CIE;
import java.util.Scanner;

public void inputCIEmarks() {
    Scanner s = new Scanner(System.in);
    System.out.println("Enter Internal Marks");
    for (int i =0 ; i<5 ; i++) {
        System.out.println("Subject" +(i+1));
        internalMarks[i] = s.nextInt();
    }
}
```

Student.java

```
package CIE;
import java.util.Scanner;

public class Student {
    protected String usn;
    protected String name;
    protected int sem;

    public void inputStudentDetails() {
        Scanner s = new Scanner(System.in)
        System.out.print("Enter USN: ");
        usn = s.nextLine();
        System.out.println("Enter Semester");
        sem = s.nextInt();
    }
}
```

Output:-

```
Enter details for Student 1:
Enter USN : 2UBECS400
Enter Name : Bhavan.A
Enter Semester : 3

Enter Internal Marks for 3 subjects:
Subject 1 : 100
Subject 2 : 50
Subject 3 : 60

Enter External Marks for 3 subjects:
Subject 1 : 100
Subject 2 : 99
Subject 3 : 100
```

Code:

CIE
–Internals.java

```
package CIE;
import java.util.Scanner;

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

    public void inputCIEmarks() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter Internal Marks for 5 subjects:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Subject " + (i + 1) + ": ");
            internalMarks[i] = s.nextInt();
        }
    }
}
```

–Student.java

```
package CIE;
import java.util.Scanner;

public class Student {
    protected String usn;
    protected String name;
    protected int sem;

    public void inputStudentDetails() {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter USN: ");
        usn = s.nextLine();
        System.out.print("Enter Name: ");
        name = s.nextLine();
        System.out.print("Enter Semester: ");
        sem = s.nextInt();
    }

    public void displayStudentDetails() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Semester: " + sem);
    }
}
```

SEE
–Externals.java

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

public class Externals extends Internals {
   protected int[] externalMarks = new int[5];
   protected int[] finalMarks = new int[5];

   public void inputSEEmarks() {
      Scanner s = new Scanner(System.in);
      System.out.println("Enter SEE Marks for 5 subjects:");
      for (int i = 0; i < 5; i++) {
         System.out.print("Subject " + (i + 1) + ": ");
         externalMarks[i] = s.nextInt();
      }
   }

   public void calculateFinalMarks() {
      for (int i = 0; i < 5; i++) {
         finalMarks[i] = internalMarks[i] + (externalMarks[i] / 2);
      }
   }

   public void displayFinalMarks() {
      displayStudentDetails();
      System.out.println("Final Marks in 5 subjects:");
      for (int i = 0; i < 5; i++) {
         System.out.println("Subject " + (i + 1) + ": " + finalMarks[i]);
      }
   }
}
```

Main.java

```
import SEE.Externals;
import java.util.Scanner;

public class Main {
   public static void main(String[] args) {
         System.out.print("Bhuvan. A\n 24BECS400\n");

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

```
        System.out.print("Enter number of students: ");
        int n = s.nextInt();
        Externals[] students = new Externals[n];

        // Input and calculate marks for each student
        for (int i = 0; i < n; i++) {
            System.out.println("\nEntering details for Student " + (i + 1) + ":");
            students[i] = new Externals();
            students[i].inputStudentDetails();
            students[i].inputCIEmarks();
            students[i].inputSEEmarks();
            students[i].calculateFinalMarks();
        }

        // Display final marks for each student
        System.out.println("\nFinal Marks of Students:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nStudent " + (i + 1) + ":");
            students[i].displayFinalMarks();
        }
    }
}
```

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

D:\24BECS400\Main>javac -d Main.java
error: no source files

D:\24BECS400\Main>javac -d Main.java
error: no source files

D:\24BECS400\Main>javac -d Main Main.java

D:\24BECS400\Main>java Main
Error: Could not find or load main class Main
Caused by: java.lang.ClassNotFoundException: Main

D:\24BECS400\Main>java Main.java
Bhuvan. A
 24BECS400
Enter number of students: 3

Entering details for Student 1:
Enter USN: 24BECS400
Enter Name: Bhuvan
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 100
Subject 2: 99
Subject 3: 98
Subject 4: 99
Subject 5: 10
Enter SEE Marks for 5 subjects:
Subject 1: 100
Subject 2: 100
Subject 3: 100
Subject 4: 100
Subject 5: 100

Entering details for Student 2:
Enter USN: 24BECS430
Enter Name: Ashwini
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 100
Subject 2: 99
Subject 3: 98
Subject 4: 97
Subject 5: 96
Enter SEE Marks for 5 subjects:
Subject 1: 99
Subject 2: 96
Subject 3: 93
```

Program 7

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

20/11/24 Week-7

~~Creating your own Exception Subclass:~~
~~Define a subclass of Exception Class which is of type~~

WAP Demonstrates a handling of a exception in Inheritance tree. Create a base class "Father" and derived class called "Son". which extends the base class.
In fathers class implement a constructor which takes the age and throws the exception "WrongAge()" when the input age is less than 0 [age <0].
In "son's class implement a constructor that uses both father and son's age and throws an exception if ~~fathers~~ fathers age ≥ son's age.

```
import java.util.Scanner;
class WrongAgeException extends Exception {
    public WrongAgeException (String message) {
        super (message);
    }
}

class SonAgeException extends Exception {
    public SonAgeException (String message) {
        super (message);
    }
}
```

```
class Father {
    private Father (int age) throws WrongAge Exception {
        if (age < 0) {
            throw new WrongAge Exception ("Wrong Age");
        }
        this.age = age;
    }
    public int getAge () {
        return age;
    }
}

class Son extends Father {
    private int sonAge
    public Son (int fatherAge, int sonAge) throws
        WrongAgeException, SonAge Exception {
        super (fatherAge);
        if (sonAge >= fatherAge) {
            throw new SonAge Exception ("Son Age
            cannot be greater than or equal to
            father's age");
        }
        this.sonAge = sonAge ();
    }
}

public class FatherSon {
    public static void main (String args[])
        while (true) {
            System.out.print ("Enter Father's age");
            Scanner sc = new Scanner (System.in);
```

```
System.out.println ("Enter Son's age:");
int SonAge = sc.nextInt ();

try {
    Son son = new Son (father age, sonAge);
    System.out.println ("Accepted Successfully");
}
catch (WrongAgeException) {
    System.out.println (e.getMessage ());
}
catch (SonAge Exception) {
    System.out.println (e.getMessage ());
}
if (input.equalsIgnoreCase ("n")) {
    break;
}
}
}
```

Output:

```
Enter Father's Age : 32
Enter Son's Age : 12

Accepted Successfully

Enter Father's Age : 18
Enter Son's Age : 20
Son's age cannot be greater than or equal to
    father's age
```

Code:

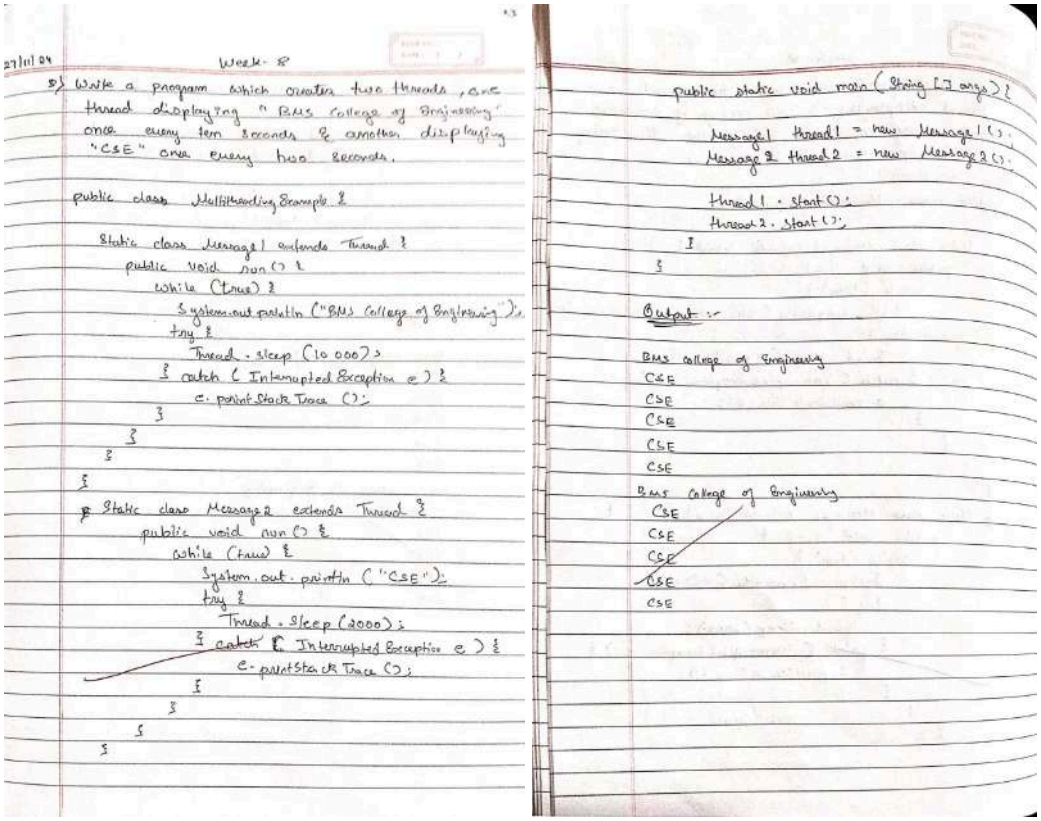
```
import java.util.Scanner;
class WrongAgeException extends Exception {
   public WrongAgeException(String message) {
      super(message);
   }
}

class SonAgeException extends Exception {
   public SonAgeException(String message) {
      super(message);
   }
}

class Father {
   private int age;
   public Father(int age) throws WrongAgeException {
      if (age < 0) {
         throw new WrongAgeException("Wrong age");
      }
      this.age = age;
   }
   public int getAge() {
      return age;
   }
}

class Son extends Father {
   private int sonAge;
   public Son(int fatherAge, int sonAge) throws WrongAgeException, SonAgeException {
      super(fatherAge);
      if (sonAge >= fatherAge) {
         throw new SonAgeException("Son's age cannot be greater than or equal to father's age");
      }
      this.sonAge = sonAge;
   }
   public int getSonAge() {
      return sonAge;
   }
}
public class FatherSon{
   public static void main(String[] args) {
      while(true){
         System.out.print("Bhuvan. A \n 24BECS400\n");
         Scanner sc = new Scanner(System.in);
         System.out.print("Enter Father's Age: ");
         int fatherAge = sc.nextInt();
```

```
            System.out.print("Enter Son's Age: ");
            int sonAge = sc.nextInt();
            try {
                Son son = new Son(fatherAge, sonAge);
                System.out.println("Accepted Succesfully");
            }
            catch (WrongAgeException e) {
                System.out.println(e.getMessage());
            }
            catch (SonAgeException e) {
                System.out.println(e.getMessage());
            }
            System.out.println("Would you like to re-enter details (Y/n)");
            String input = sc.next();
            if (input.equalsIgnoreCase("n")) {
                break;
            }
        }
    }
}
```

```
D:\24BECS400\week7>javac FatherSon.java

D:\24BECS400\week7>java FatherSon
Bhuvan. A
 24BECS400
Enter Father's Age: 32
Enter Son's Age: 12
Accepted Succesfully
Would you like to re-enter details (Y/n)
y
Bhuvan. A
 24BECS400
Enter Father's Age: 18
Enter Son's Age: 20
Son's age cannot be greater than or equal to father's age
Would you like to re-enter details (Y/n)
y
Bhuvan. A
 24BECS400
Enter Father's Age: 0
Enter Son's Age: 0
Son's age cannot be greater than or equal to father's age
Would you like to re-enter details (Y/n)
y
Bhuvan. A
 24BECS400
Enter Father's Age: -1
Enter Son's Age: 0
Wrong age
Would you like to re-enter details (Y/n)
n
```

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.



Week- 8

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

```
public class MultiThreadingExample {

    Static class Message1 extends Thread {
        public void run() {
            while (true) {
                System.out.println("BMS College of Engineering");
                try {
                    Thread.sleep(10 000);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        }
    }
    Static class Message2 extends Thread {
        public void run() {
            while (true) {
                System.out.println("CSE");
                try {
                    Thread.sleep(2000);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        }
    }

    public static void main(String[] args) {
        Message1 thread1 = new Message1();
        Message2 thread2 = new Message2();
        thread1.start();
        thread2.start();
    }
}
```

Output :-

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

Code:

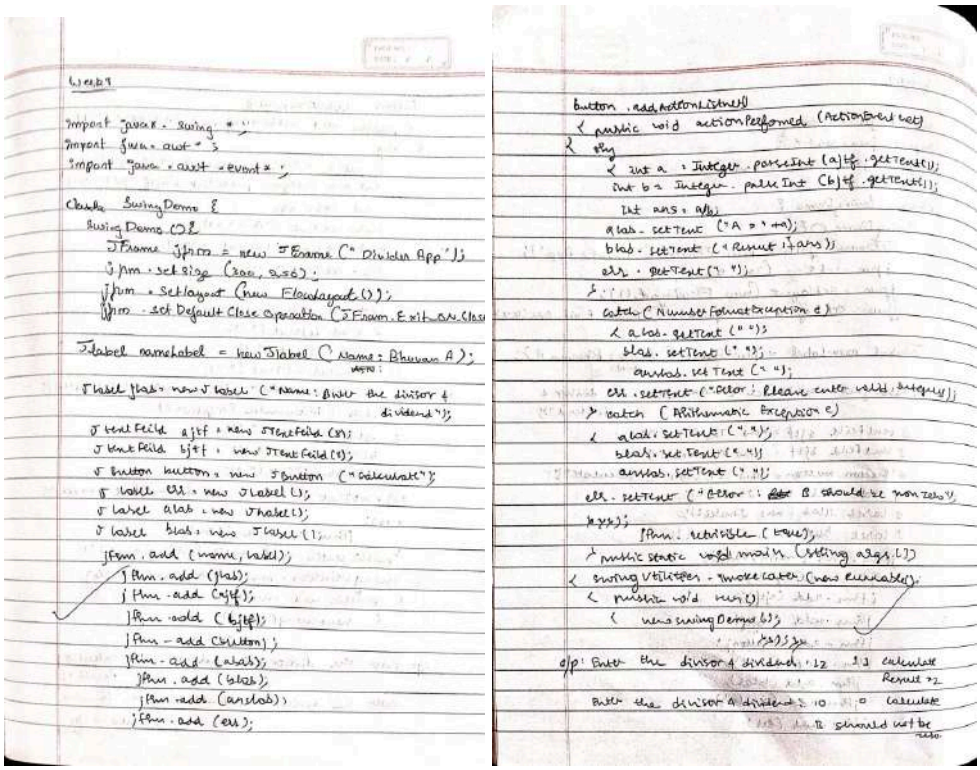
```
public class MultiThreadExample {

    static class Message1 extends Thread {
        public void run() {
            while (true) {
                System.out.println("BMS College of Engineering");
                try {
                    Thread.sleep(10000);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        }
    }


    static class Message2 extends Thread {
        public void run() {
```

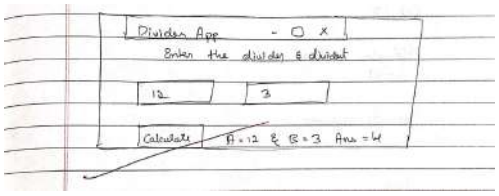
```
        while (true) {
            System.out.println("CSE");
            try {
                Thread.sleep(2000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
  }

  public static void main(String[] args) {
    Message1 thread1 = new Message1();
    Message2 thread2 = new Message2();

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

```
D:\24BECS400\week8>java Deadlock
RacingThread entered B.bar
MainThread entered A.foo
RacingThread trying to call A.last()
MainThread trying to call B.last()
^C
D:\24BECS400\week8>

D:\24BECS400\week8>javac MultiThreadExample.java

D:\24BECS400\week8>java MultiThreadExample
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
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
```

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.



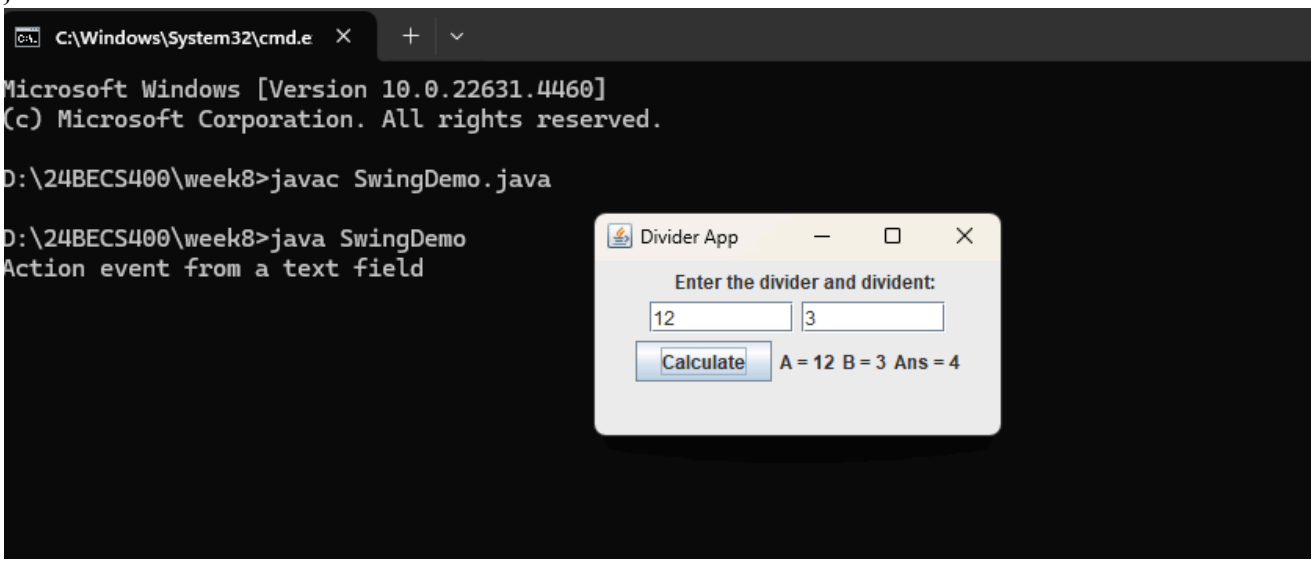


Code:

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

class SwingDemo {
   SwingDemo() {
      // create jframe container
      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(275, 150);
      jfrm.setLayout(new FlowLayout());
      // to terminate on close
```

```
jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
// text label
JLabel jlab = new JLabel("Enter the divider and divident:");
// add text field for both numbers
JTextField ajtf = new JTextField(8);
JTextField bjtf = new JTextField(8);
// calc button
JButton button = new JButton("Calculate");
// labels
JLabel err = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();

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

         err.setText("Enter Only Integers!");
      } catch (ArithmeticException e) {
         alab.setText("");
```

```
                blab.setText("");
                anslab.setText("");
                err.setText("B should be NON zero!");
            }
        }
    });
    // display frame
    jfrm.setVisible(true);
  }

  public static void main(String args[]) {
    // create frame on event dispatching thread
    SwingUtilities.invokeLater(new Runnable() {
        public void run() {
            new SwingDemo();
        }
    });
  }
}
```



Program 10

Demonstrate Inter process Communication and deadlock.

```
Week-10

Class A {
  Synchronized void foo (B b)
  {
    String name = Thread.currentThread().getName();
    SOP (name + " entered A.foo");

  try {
    Thread.sleep(1000);
  }
  catch (Exception e) {
    SOP ("A Interrupted");
  }
  SOP (name + " trying to call B.last()");
    b.last();
  }
  void last()
  {
    SOP ("Inside A.last");
  }
}
Class B {
  Synchronized void bar (A a)
  {
    String name = Thread.currentThread().getName();
    SOP (name + " entered B.bar");
try {
    Thread.sleep(1000);
  }
  catch (Exception e) {
    SOP ("B Interrupted");
    a.last();
  }
  Void last() {
    SOP ("Inside B.last");
  }
}
Class Deadlock Implements Runnable {
    A a = new A();
    B b = new B();

Deadlock() {
  Thread.currentThread().setName("Main");
  Thread t = new Thread(this, "Racing");
  t.start();
    a.foo(b);
    SOP ("Back in main");
  }
public void run() {
    b.bar(a);
    SOP ("Back in other thread");
  }
public static void main (String args[]) {
    SOP ("Name : Bhuvan");
    new Deadlock();
  }
```

```
10) Demonstrate Inter process Communication & deadlock

P. Output:

Put: 0

Intimate Consumer

Producer waiting
Get: 0

Intimate Producer
Put: 1

Intimate Consumer

Output:-

Racing Thread entered B.bar
Main Thread entered A.foo
Racing Thread trying to call A.last()
Main Thread trying to call B.last()
```

Seen
9/11/24

Code:

Deadlock

```
class A {
   synchronized void foo(B b) {
      String name = Thread.currentThread().getName();
      System.out.println(name + " entered A.foo");

      try {
         Thread.sleep(1000);
      } catch (Exception e) {
         System.out.println("A Interrupted");
      }

      System.out.println(name + " trying to call B.last()");
      b.last();
   }

   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 B.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();

        // Ensure that main thread always calls a.foo(b)
        synchronized (a) {  // Lock a before b to avoid circular waiting
            a.foo(b); // get lock on a in this
        }

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

    public void run() {
        // Ensure that the other thread always calls b.bar(a)
        synchronized (b) {  // Lock b before a to avoid circular waiting
            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();
    }
}
```

```
D:\24BECS400\week8>javac Deadlock.java

D:\24BECS400\week8>java Deadlock
RacingThread entered B.bar
MainThread entered A.foo
RacingThread trying to call A.last()
MainThread trying to call B.last()
```


```
D:\24BECS400\week8>javac Deadlock.java

D:\24BECS400\week8>java Deadlock
RacingThread entered B.bar
MainThread entered A.foo
RacingThread trying to call A.last()
MainThread trying to call B.last()
```

Process Communication

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

   synchronized int get() {
      while (!valueSet) {
         try {
            System.out.println("\nConsumer waiting");
            wait();  // Consumer waits if value is not set
         } catch (InterruptedException e) {
            System.out.println("InterruptedException caught in get()");
            Thread.currentThread().interrupt();  // Re-interrupt the thread
         }
      }
      System.out.println("Got: " + n);
      valueSet = false;  // Mark the value as consumed
      System.out.println("\nIntimate Producer");
      notify();  // Notify producer to put new value
      return n;
   }

   synchronized void put(int n) {
      while (valueSet) {
         try {
            System.out.println("\nProducer waiting");
            wait();  // Producer waits if value has already been set
         } catch (InterruptedException e) {
            System.out.println("InterruptedException caught in put()");
            Thread.currentThread().interrupt();  // Re-interrupt the thread
         }
      }
      this.n = n;
      valueSet = true;  // Mark the value as produced
      System.out.println("Put: " + n);
      System.out.println("\nIntimate Consumer");
      notify();  // Notify consumer that value is available
   }
}
```

```
class Producer implements Runnable {
   Q q;
   private static final int MAX_ITEMS = 15;

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

   public void run() {
      int i = 0;
      while (i < MAX_ITEMS) {  // Produce only up to MAX_ITEMS
         q.put(i++);
      }
   }
}

class Consumer implements Runnable {
   Q q;
   private static final int MAX_ITEMS = 15;

   Consumer(Q q) {
      this.q = q;
      new Thread(this, "Consumer").start();
   }
   public void run() {
      int i = 0;
      while (i < MAX_ITEMS) {  // Consume only up to MAX_ITEMS
         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.");
    }
}
```

```
Producer waiting
Got: 11

Intimate Producer
Consumed: 11
Put: 12

Intimate Consumer

Producer waiting
Got: 12

Intimate Producer
Consumed: 12
Put: 13

Intimate Consumer

Producer waiting
Got: 13

Intimate Producer
Consumed: 13
Put: 14

Intimate Consumer
Got: 14

Intimate Producer
Consumed: 14

D:\24BECS400\week8>
```