M.Sc. (Five Year Integrated) in Computer Science (Artificial Intelligence & Data Science)

Second Semester

Laboratory Record

21-805-0207: JAVA PROGRAMMING LAB

Submitted in partial fulfillment
of the requirements for the award of degree in
Master of Science (Five Year Integrated)
in Computer Science (Artificial Intelligence & Data Science) of
Cochin University of Science and Technology (CUSAT)
Kochi

Submitted by

LANA ANVAR
(80522012)

DEPARTMENT OF COMPUTER SCIENCE
COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY (CUSAT)
KOCHI-682022

OCTOBER 2023

DEPARTMENT OF COMPUTER SCIENCE

COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY (CUSAT)

KOCHI, KERALA-682022

This is to certify that the software laboratory record for ***21-805-0207: JAVA Programming Lab*** *is a record of work carried out by* ***LANA ANVAR(80522012)****, in partial fulfillment of the requirements for the award of degree in* ***Master of Science (Five Year Integrated)*** *in* ***Computer Science (Artificial Intelligence & Data Science)*** *of Cochin University of Science and Technology (CUSAT), Kochi. The lab record has been approved as it satisfies the academic requirements in respect of the first semester laboratory prescribed for the Master of Science (Five Year Integrated) in Computer Science degree.*

Faculty Member in-charge

Dr.Shailesh S.	Dr. Madhu S.Nair
Assistant Professor	Professor and Head
Department of Computer Science	Department of Computer Science
CUSAT	CUSAT

Kochi

11 – 10 – 2023

Table of Contents

PRINT PRIME NUMBERS

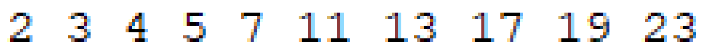
AIM

Write a program to print first ‘n’ prime numbers (read ‘n’ as a command line argument)

PROGRAM

```
package com.mycompany.primenum;

/**
 *
 * @author lanaa
 */
public class PrimeNum {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        int n=Integer.parseInt(args[0]);
        int count=0;
        int num=2;

        while(count<n)
        {
            if(CheckPrime(num)==true)
            {
                System.out.print(num+" ");
                count++;
            }
            num++;
        }
    }

    public static boolean CheckPrime(int num)
    {
        for(int i=2; i<num/2; i++)
        {
            if(num%i==0)
            {
                return false;
            }
        }
        return true;
    }

}
```

SAMPLE INPUT-OUTPUT



```
2  3  4  5  7  11  13  17  19  23
```

PRINT STRING FROM COMMAND LINE ARGUMENT

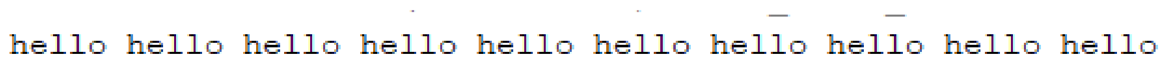
AIM

Write a program to read a number ‘n’ and a string ‘str’ as a command line argument and print ‘str’ n times.

PROGRAM

```
package com.mycompany.printstr;

/**
 *
 * @author lanaa
 */
public class PrintStr {

    public static void main(String[] args) {
        int n=Integer.parseInt(args[0]);
        String str=args[1];

        for(int i=0; i<n; i++)
        {
            System.out.print(str+" ");
        }
    }
}
```

SAMPLE INPUT-OUTPUT



```
hello hello hello hello hello hello hello hello hello hello
```

STUDENT MARKS

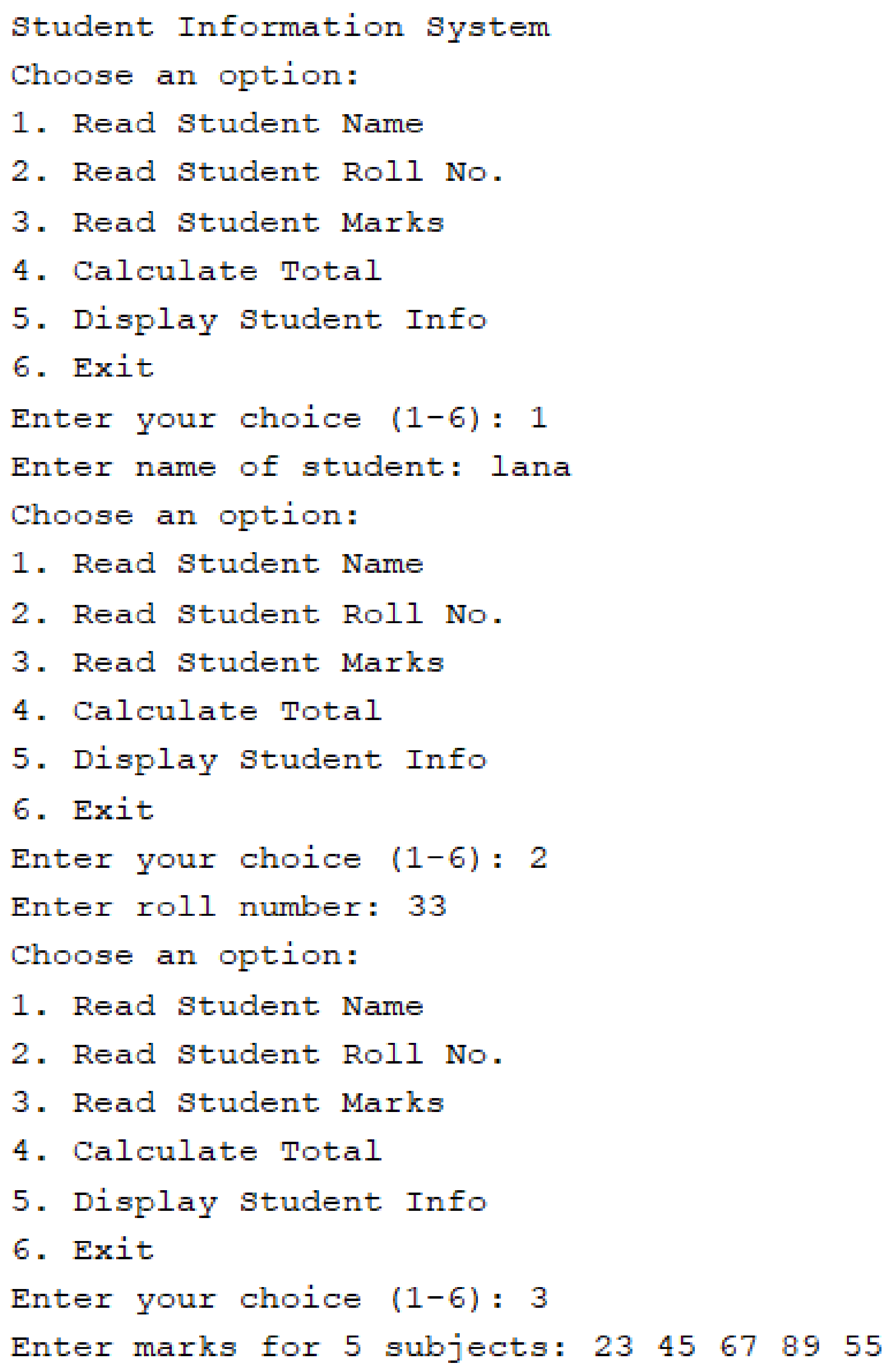
AIM

Program to write a class Student having members ‘name’, ‘roll number’, ‘5 subject marks’ and ‘total’. Provide methods for a) Initializing name, roll number and marks. b) Calculate the total c) Get back the total d) Print the details Create 2 Student objects and print the details of the student with a greater tota

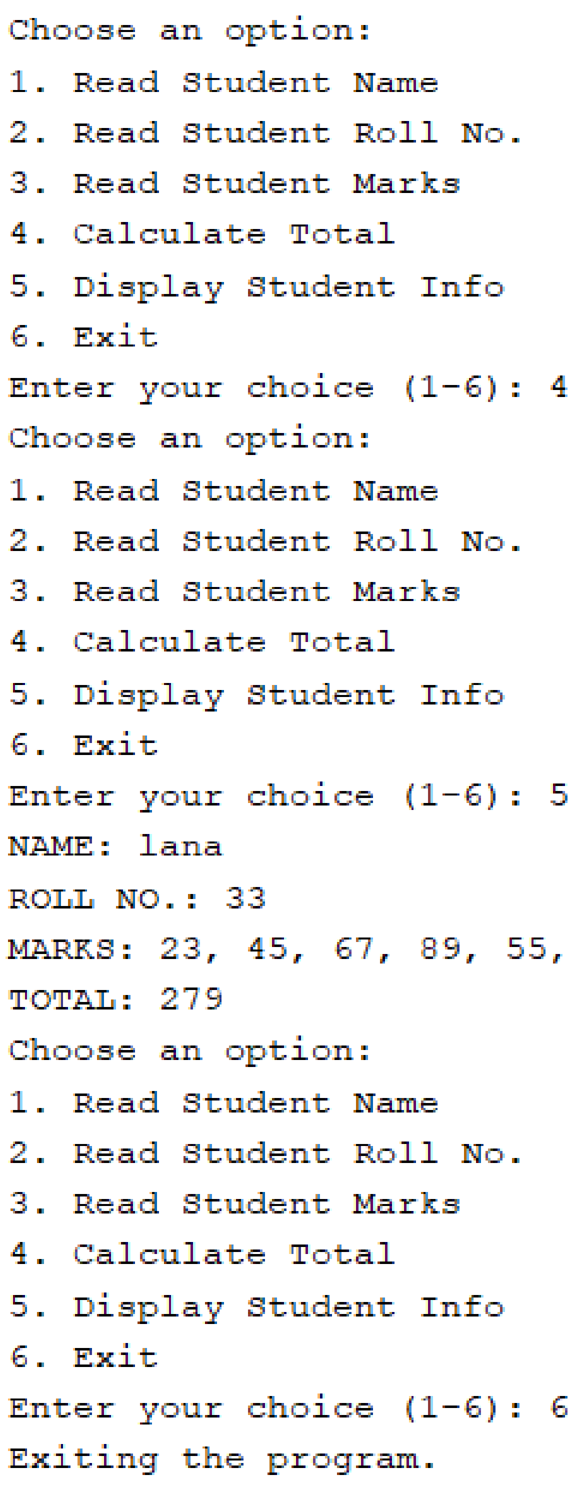
PROGRAM

```
import java.util.Scanner;

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

        System.out.println("Student Information System");

        while (true) {
            System.out.println("Choose an option:");
            System.out.println("1. Read Student Name");
            System.out.println("2. Read Student Roll No.");
            System.out.println("3. Read Student Marks");
            System.out.println("4. Calculate Total");
            System.out.println("5. Display Student Info");
            System.out.println("6. Exit");
            System.out.print("Enter your choice (1-6): ");
            int choice = scanner.nextInt();
            scanner.nextLine(); // Consume the newline character

            switch (choice) {
                case 1:
                    studentHandler.readName();
                    break;

                case 2:
                    studentHandler.readRollno();
                    break;

                case 3:
                    studentHandler.readMarks();
                    break;

                case 4:
                    studentHandler.calculateTotal();
                    break;

                case 5:
                    studentHandler.displayStudentInfo();
                    break;

                case 6:
                    System.out.println("Exiting the program.");
                    scanner.close();
                    System.exit(0);

                default:
                    System.out.println("Invalid choice.
                    Please enter a number between 1 and 6.");
            }
        }
    }
}

class StudentInfoHandler {
    private String name;
    private int rollno;
    private int[] marks = new int[5];
    private int total;

    public void readName() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter name of student: ");
        this.name = sc.next();
    }

    public void readRollno() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter roll number: ");
        this.rollno = sc.nextInt();
    }
    public void readMarks() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter marks for 5 subjects: ");
        for (int i = 0; i < 5; i++) {
            this.marks[i] = sc.nextInt();
        }
    }

    public void calculateTotal() {
        this.total = 0;
        for (int i = 0; i < 5; i++) {
            this.total += this.marks[i];
        }
    }

    public void displayStudentInfo() {
        System.out.println("NAME: " + this.name);
        System.out.println("ROLL NO.: " + this.rollno);
        System.out.print("MARKS: ");
        for (int i = 0; i < 5; i++) {
            System.out.print(this.marks[i] + ", ");
        }
        System.out.println();
        System.out.println("TOTAL: " + this.total);
    }
}
```

SAMPLE INPUT-OUTPUT



```
Student Information System
Choose an option:
1. Read Student Name
2. Read Student Roll No.
3. Read Student Marks
4. Calculate Total
5. Display Student Info
6. Exit
Enter your choice (1-6): 1
Enter name of student: lana
Choose an option:
1. Read Student Name
2. Read Student Roll No.
3. Read Student Marks
4. Calculate Total
5. Display Student Info
6. Exit
Enter your choice (1-6): 2
Enter roll number: 33
Choose an option:
1. Read Student Name
2. Read Student Roll No.
3. Read Student Marks
4. Calculate Total
5. Display Student Info
6. Exit
Enter your choice (1-6): 3
Enter marks for 5 subjects: 23 45 67 89 55
```



```
Choose an option:
1. Read Student Name
2. Read Student Roll No.
3. Read Student Marks
4. Calculate Total
5. Display Student Info
6. Exit
Enter your choice (1-6): 4
Choose an option:
1. Read Student Name
2. Read Student Roll No.
3. Read Student Marks
4. Calculate Total
5. Display Student Info
6. Exit
Enter your choice (1-6): 5
NAME: lana
ROLL NO.: 33
MARKS: 23, 45, 67, 89, 55,
TOTAL: 279
Choose an option:
1. Read Student Name
2. Read Student Roll No.
3. Read Student Marks
4. Calculate Total
5. Display Student Info
6. Exit
Enter your choice (1-6): 6
Exiting the program.
```

COMPLEX OPERATIONS

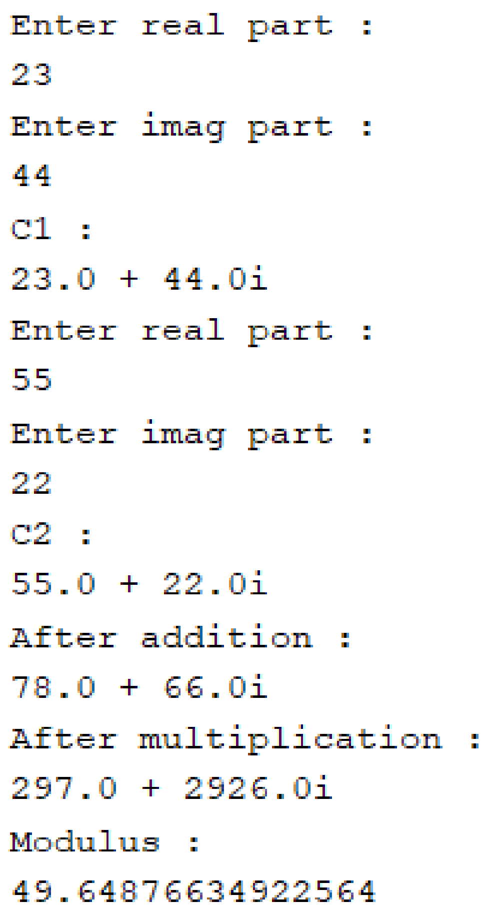
AIM

Write a program to create a class Complex have two members, real and imaginary and methods to initialize and print the complex number. Create another class ComplexOperations and provide static methods to add, multiply and get the modulus

PROGRAM

```
package com.mycompany.complex;
import java.util.Scanner;

/**
 *
 * @author lanaa
 */
public class complexMain {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {

        Scanner sc=new Scanner(System.in);
        System.out.println("Enter real part : ");
        double real1= sc.nextDouble();
        System.out.println("Enter imag part : ");
        double imag1= sc.nextDouble();

        Complex C1=new Complex();
        C1.setReal(real1);
        C1.getReal();

        C1.setImag(imag1);
        C1.getImag();

        System.out.println("C1 : ");
        C1.printComplex();

        Complex C2=new Complex();
        System.out.println("Enter real part : ");
        double real2= sc.nextDouble();
        System.out.println("Enter imag part : ");
        double imag2= sc.nextDouble();


        C2.setReal(real2);
        C2.getReal();

        C2.setImag(imag2);
        C2.getImag();
        System.out.println("C2 : ");

        C2.printComplex();

        Complex C3=complexOperations.add(C1,C2);
        System.out.println("After addition : ");
        C3.printComplex();

        Complex C4= complexOperations.multiply(C1, C2);
        System.out.println("After multiplication : ");
        C4.printComplex();

        double modulus=complexOperations.modulus(C1);
        System.out.println("Modulus : ");
        System.out.print(modulus);
    }

}
```

SAMPLE INPUT-OUTPUT



```
Enter real part :
23
Enter imag part :
44
C1 :
23.0 + 44.0i
Enter real part :
55
Enter imag part :
22
C2 :
55.0 + 22.0i
After addition :
78.0 + 66.0i
After multiplication :
297.0 + 2926.0i
Modulus :
49.64876634922564
```

AREA AND VOLUME OF SHAPES

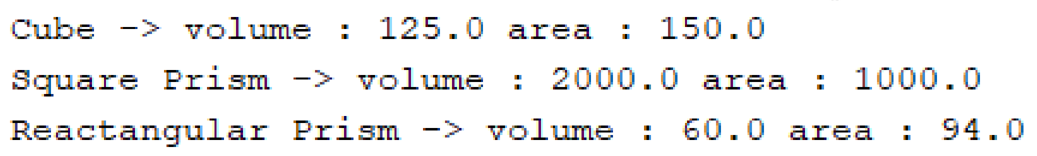
AIM

Write a program to create a class Box with data members length, breadth, height, area and volume. Provide 3 constructors having one parameter (for cube), two parameters (for square prism) three parameters (rectangular prism). Also provide functions to calculate area and volume. Find the area of a cube, a square prism and a rectangular prism using the above class.

PROGRAM

```
package com.mycompany.box;

/**
 *
 * @author lanaa
 */
public class Box {

    double l, b, h, area, volume;

    public Box()
    {
        l=0;
        b=0;
        h=0;
        area=0;
        volume=0;
    }
    public Box(double l, double b, double h)
    {
        this.l=l;
        this.b=b;
        this.h=h;
    }
    public Box(double s1, double s2)
    {
        this.l=s1;
        this.b=s1;
        this.h=s2;
    }
    public Box(double s)
    {
        this.l=s;
        this.b=s;
        this.h=s;
    }
    public double area()
    {
        this.area=2*((l*b)+(b*h)+(h*l));
        return area;
    }
    public double volume()
    {
        this.volume=l*b*h;
        return volume;
    }

}
package com.mycompany.box;

/**
 *
 * @author lanaa
 */
public class BoxMain {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        Box b1=new Box(5);
        System.out.println("Cube -> volume : "+b1.volume()+
        "
        area : "+b1.area());

        Box b2=new Box(10,20);
        System.out.println("Square Prism -> volume : "+b2.volume()+
        " area : "+b2.area());

        Box b3=new Box(3,4,5);
        System.out.println("Reactangular Prism -> volume : "+b3.volume()+
        " area : "+b3.area());
    }

}
```

SAMPLE INPUT-OUTPUT



```
Cube -> volume : 125.0 area : 150.0
Square Prism -> volume : 2000.0 area : 1000.0
Reactangular Prism -> volume : 60.0 area : 94.0
```

AREA AND VOLUME OF RECTANGLE

AIM

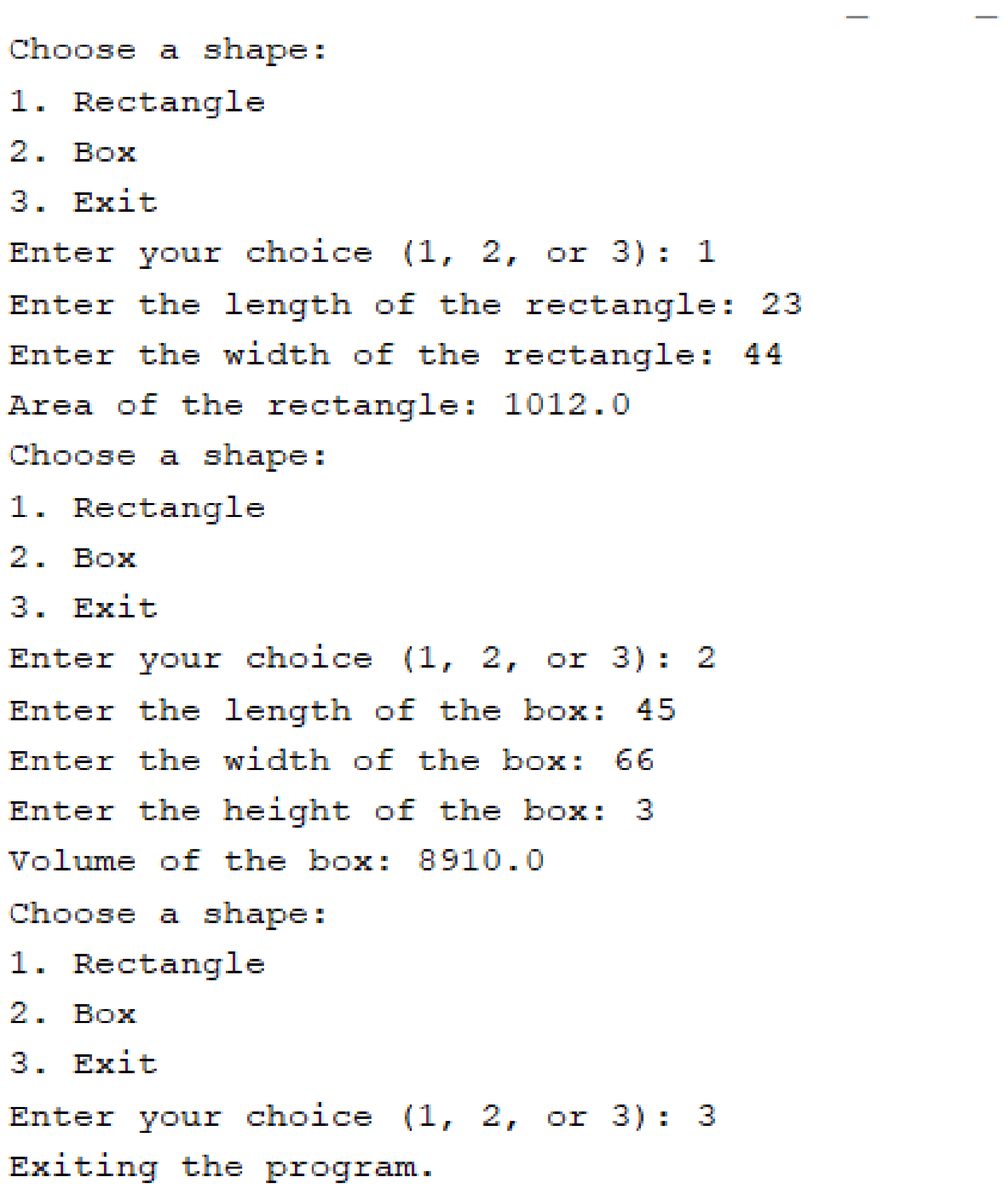
Write a program to create a class called Rectangle with members length, breadth and area. Provide functions to find area and get back the area. Create a new class Box by extending Rectangle class add two new members, height and volume and also new functions to calculate and get back the volume.

PROGRAM

```
package com.mycompany.rectangle;

/**
 *
 * @author lanaa
 */
public class Rectangle {

    protected double length;
    protected double breadth;
    protected double area;

    public Rectangle()
    {
        this.length=0;
        this.breadth=0;
        this.area=0;
    }
    public Rectangle(double l, double b)
    {
        this.length=l;
        this.breadth=b;
    }
    public double getArea()
    {
        return length*breadth;
    }
}
package com.mycompany.rectangle;

/**
```

```
 *
 * @author lanaa
 */
public class Box extends Rectangle {
    protected double height;
    protected double vol;

    public Box()
    {
        super();
        this.height=0;
    }

    public Box(double l, double b, double h)
    {
        super(l,b);
        this.height=h;
    }
    public double getVol()
    {
        return length*breadth*height;
    }
}
import java.util.Scanner;

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

        while (true) {
            System.out.println("Choose a shape:");
            System.out.println("1. Rectangle");
            System.out.println("2. Box");
            System.out.println("3. Exit");
            System.out.print("Enter your choice (1, 2, or 3): ");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter the length of the
                    rectangle: ");
                    double length = scanner.nextDouble();
                    System.out.print("Enter the width of the
                    rectangle: ");
                    double width = scanner.nextDouble();

                    Rectangle r = new Rectangle(length, width);
                    System.out.println("Area of the rectangle: " +
                    r.getArea());
                    break;

                case 2:
                    System.out.print("Enter the length of the
                    box: ");
                    double boxLength = scanner.nextDouble();
                    System.out.print("Enter the width of the
                    box: ");
                    double boxWidth = scanner.nextDouble();
                    System.out.print("Enter the height of the
                    box: ");
                    double boxHeight = scanner.nextDouble();

                    Box b = new Box(boxLength, boxWidth, boxHeight);
                    System.out.println("Volume of the box: " +
                    b.getVol());
                    break;

                case 3:
                    System.out.println("Exiting the program.");
                    scanner.close();
                    System.exit(0);

                default:
                    System.out.println("Invalid choice.
                    Please enter 1, 2, or 3.");
            }
        }
    }
}
```

SAMPLE INPUT-OUTPUT



```
Choose a shape:
1. Rectangle
2. Box
3. Exit
Enter your choice (1, 2, or 3): 1
Enter the length of the rectangle: 23
Enter the width of the rectangle:  44
Area of the rectangle: 1012.0
Choose a shape:
1. Rectangle
2. Box
3. Exit
Enter your choice (1, 2, or 3): 2
Enter the length of the box:  45
Enter the width of the box:  66
Enter the height of the box:  3
Volume of the box: 8910.0
Choose a shape:
1. Rectangle
2. Box
3. Exit
Enter your choice (1, 2, or 3): 3
Exiting the program.
```

BANK TRANSACTIONS

AIM

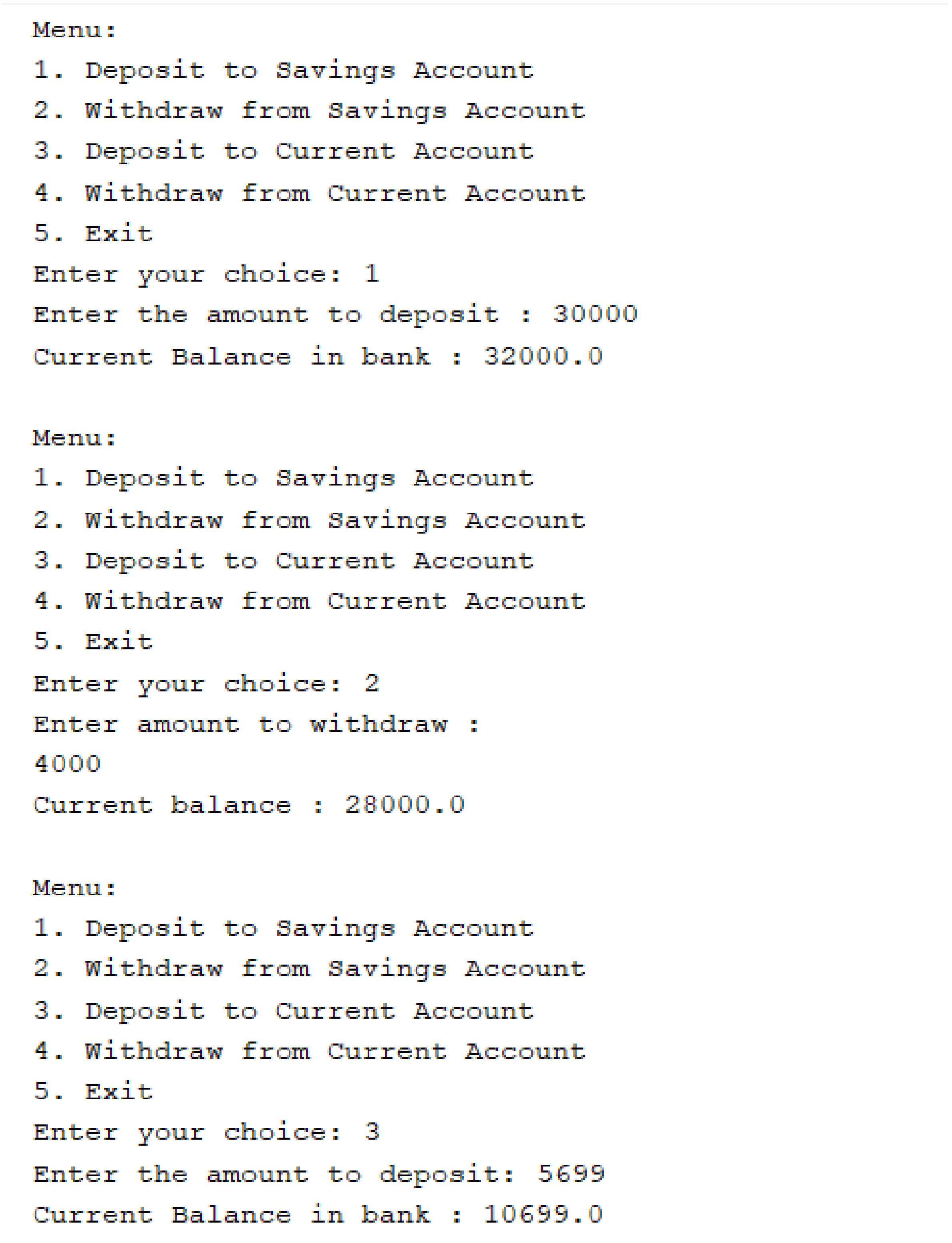
Write a program to create an abstract base class Account with 3 members account holder name, account number and balance amount. Provide constructor to initialize data members, function to deposit cash to account and an abstract function, withdrawal. Create two child classes Saving Account and Current Account of Account class. Override abstract function withdrawal in child classes as per the criteria, for savings maintain a minimum balance 1000 and for current account, one can withdraw 5overdraft amount. Illustrate the above as a menu driven program.

PROGRAM

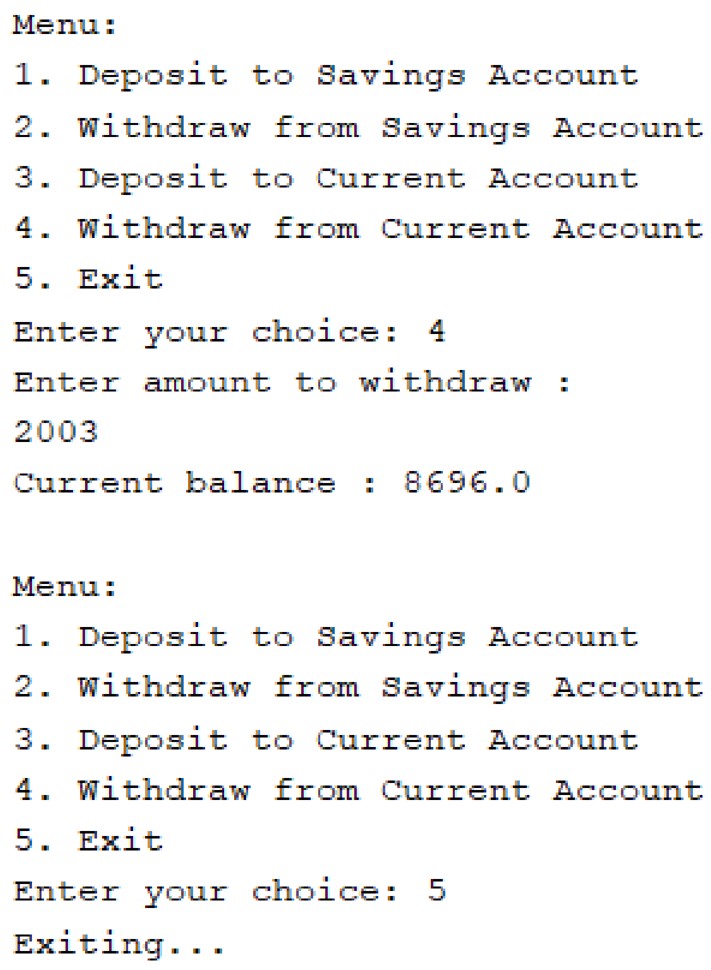
```
package com.mycompany.account;
import java.util.Scanner;

/**
 *
 * @author lanaa
 */
abstract public class Account {

    String name;
    String accNo;
    double bal;


    public Account(String name, String accNo, double bal)
    {
        this.name=name;
        this.accNo=accNo;
        this.bal=bal;
    }
    public void deposit(double b)
    {
        this.bal=this.bal+b;
        System.out.println("Current Balance in bank : "+bal);
    }
    abstract void withdrawal();
}

class SavingAccount extends Account
```

```
{
    public SavingAccount(String name, String accNo, double bal)
    {
        super(name, accNo, bal);
    }
    @Override
    public void withdrawal()
    {
        Scanner sc=new Scanner(System.in);

        System.out.println("Enter amount to withdraw : ");
        double w=sc.nextDouble();

        if(this.bal<1000)
        {
            System.out.println("Balance is less than Rs. 1000.
            Withdrawal not possible. ");
        }
        else
        {
            this.bal=this.bal-w;
        }

        System.out.println("Current balance : "+bal);
    }
}

class CurrentAccount extends Account
{
    public CurrentAccount(String name, String accNo, double bal)
    {
        super(name, accNo, bal);
    }
    @Override
    public void withdrawal()
    {
        Scanner sc=new Scanner(System.in);

        System.out.println("Enter amount to withdraw : ");
        double w=sc.nextDouble();
        if(bal<w)
        {
            System.out.println("Balance is less than Rs. 1000.
            Withdrawal not possible. Overdraft transaction on the way. ");
            w=0.05*bal;
            bal=bal-w;
        }
        else
        {
            bal=bal-w;
        }

        System.out.println("Current balance : "+bal);

    }
}
package com.mycompany.account;
import java.util.Scanner;

/**
 *
 * @author lanaa
 */
public class AccountMain {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int choice;

        // Create savings account
        SavingAccount s1 = new SavingAccount("Lana", "SAV001", 2000);

        // Create current account
        CurrentAccount c1 = new CurrentAccount("Amal", "CUR001", 5000);

        do {
            System.out.println("Menu:");
            System.out.println("1. Deposit to Savings Account");
            System.out.println("2. Withdraw from Savings Account");
            System.out.println("3. Deposit to Current Account");
            System.out.println("4. Withdraw from Current Account");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            choice = sc.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter the amount to deposit : ");
                    double savingDeposit = sc.nextDouble();
                    s1.deposit(savingDeposit);
                    break;
                case 2:
                    s1.withdrawal();
                    break;
                case 3:
                    System.out.print("Enter the amount to deposit: ");
                    double currentDeposit = sc.nextDouble();
                    c1.deposit(currentDeposit);
                    break;
                case 4:
                    c1.withdrawal();
                    break;
                case 5:
                    System.out.println("Exiting...");
                    break;
                default:
                    System.out.println("Invalid choice. Please try again.");
            }

            System.out.println();
        } while (choice != 5);

        sc.close();
    }


}
```

SAMPLE INPUT-OUTPUT



```
Menu:
1. Deposit to Savings Account
2. Withdraw from Savings Account
3. Deposit to Current Account
4. Withdraw from Current Account
5. Exit
Enter your choice: 1
Enter the amount to deposit : 30000
Current Balance in bank : 32000.0

Menu:
1. Deposit to Savings Account
2. Withdraw from Savings Account
3. Deposit to Current Account
4. Withdraw from Current Account
5. Exit
Enter your choice: 2
Enter amount to withdraw :
4000
Current balance : 28000.0

Menu:
1. Deposit to Savings Account
2. Withdraw from Savings Account
3. Deposit to Current Account
4. Withdraw from Current Account
5. Exit
Enter your choice: 3
Enter the amount to deposit: 5699
Current Balance in bank : 10699.0
```



```
Menu:
1. Deposit to Savings Account
2. Withdraw from Savings Account
3. Deposit to Current Account
4. Withdraw from Current Account
5. Exit
Enter your choice: 4
Enter amount to withdraw :
2003
Current balance : 8696.0

Menu:
1. Deposit to Savings Account
2. Withdraw from Savings Account
3. Deposit to Current Account
4. Withdraw from Current Account
5. Exit
Enter your choice: 5
Exiting...
```

AREA AND VOLUME OF 3D-SHAPES

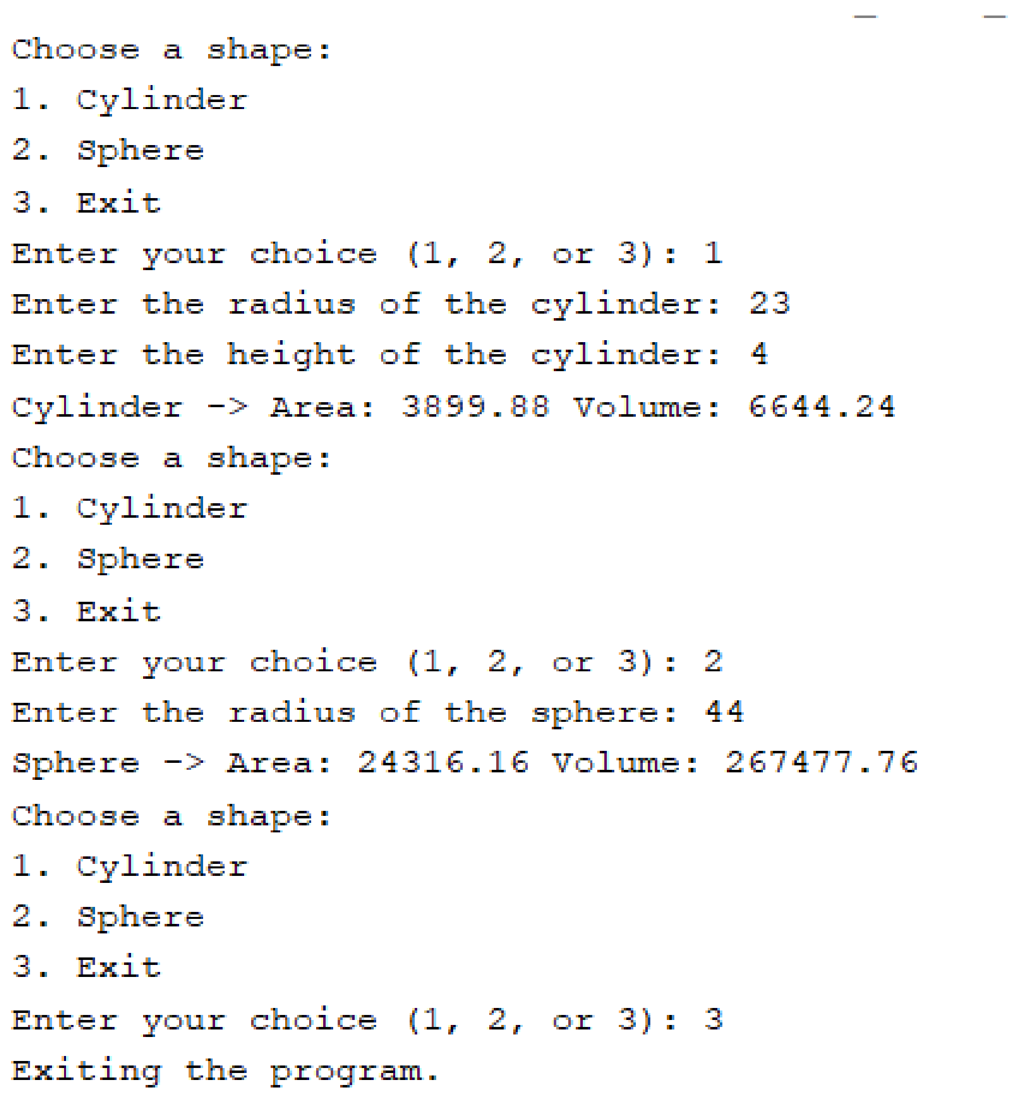
AIM

Write a program to create an interface, 3DShapes with methods printVolume() and printArea(), which prints the Volume and Area respectively. Create classes Cylinder and Sphere by implementing 3DShapes interface. Using these child classes calculate the print volume and area of a cylinder and sphere.

PROGRAM

```
package com.mycompany.shapes;

/**
 *
 * @author cusat
 */
interface Shapes {



    double printVolume();
    double printArea();
}

class Cylinder implements Shapes
{
    double rad;
    double height;

    public Cylinder()
    {
        rad=0;
        height=0;
    }
    public Cylinder(double num1, double num2)
    {
        rad=num1;
        height=num2;
    }
    @Override
    public double printVolume(){
        return 3.14*rad*rad*height;
    }
    public double printArea(){
        return 2*3.14*rad*(rad+height);
    }
}

class Sphere implements Shapes
{
    double rad;

    public Sphere()
    {
        rad=0;
    }
    public Sphere(double num1)
    {
        rad=num1;
    }
    public double printVolume()
    {
        return (4/3)*3.14*rad*rad*rad;
    }
    public double printArea()
    {
        return 4*3.14*rad*rad;
    }

}

import java.util.Scanner;

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

        while (true) {
            System.out.println("Choose a shape:");
            System.out.println("1. Cylinder");
            System.out.println("2. Sphere");
            System.out.println("3. Exit");
            System.out.print("Enter your choice (1, 2, or 3): ");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter the radius of the
                    cylinder: ");
                    double cylinderRadius = scanner.nextDouble();
                    System.out.print("Enter the height of the
                    cylinder: ");
                    double cylinderHeight = scanner.nextDouble();

                    Cylinder cylinder = new Cylinder(cylinderRadius,
                    cylinderHeight);
                    double cylVol = cylinder.printVolume();
                    double cylArea = cylinder.printArea();

                    System.out.println("Cylinder -> Area: " +
                    cylArea + " Volume: " + cylVol);
                    break;

                case 2:
                    System.out.print("Enter the radius of the
                    sphere: ");
                    double sphereRadius = scanner.nextDouble();

                    Sphere sphere = new Sphere(sphereRadius);
                    double sphVol = sphere.printVolume();
                    double sphArea = sphere.printArea();

                    System.out.println("Sphere -> Area: " + sphArea +
                    " Volume: " + sphVol);
                    break;

                case 3:
                    System.out.println("Exiting the program.");
                    scanner.close();
                    System.exit(0);

                default:
                    System.out.println("Invalid choice. Please enter
                    1, 2, or 3.");
            }
        }
    }
}
```

SAMPLE INPUT-OUTPUT



```
Choose a shape:
1. Cylinder
2. Sphere
3. Exit
Enter your choice (1, 2, or 3): 1
Enter the radius of the cylinder: 23
Enter the height of the cylinder: 4
Cylinder -> Area: 3899.88 Volume: 6644.24
Choose a shape:
1. Cylinder
2. Sphere
3. Exit
Enter your choice (1, 2, or 3): 2
Enter the radius of the sphere: 44
Sphere -> Area: 24316.16 Volume: 267477.76
Choose a shape:
1. Cylinder
2. Sphere
3. Exit
Enter your choice (1, 2, or 3): 3
Exiting the program.
```

EMPLOYEE SALARY CALCULATION

AIM

Write a program to create a class Employee with data members name, code and basic pay and with functions to initialize and print information. Create an interface Salary with a function salary calculation. By inheriting the Employee class and Salary Interface create a new class SalarySlip which override the salary calculation method to calculate the net salary of an employee from basic pay. Provide a function to print the Salary Slip of the employee in SalarySlip class.

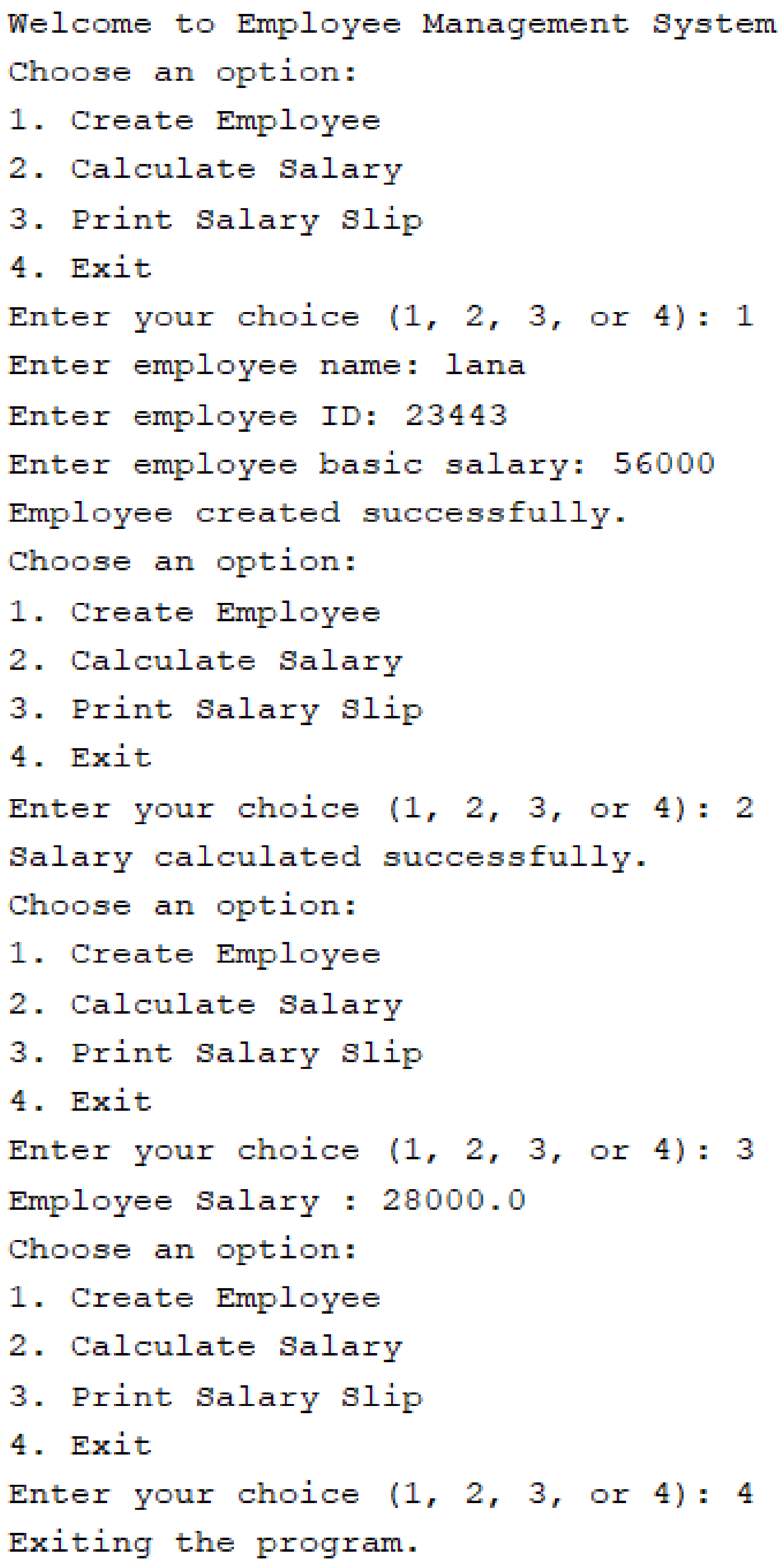
PROGRAM

```
package com.mycompany.employee;

/**
 *
 * @author lanaa
 */
class Employee {

    protected String name;
    protected String code;
    protected double basic_pay;

    public Employee(String name, String code, double basic_pay)
    {
        this.name=name;
        this.code=code;
        this.basic_pay=basic_pay;
    }
    public void print_info()
    {
        System.out.println("Employee Name : "+name);
        System.out.println("Employee code : "+code);
        System.out.println("Basic Pay : "+basic_pay);
    }
}

interface Salary{
    void SalaryCalc();
}
```

```
class SalarySlip extends Employee implements Salary{
    double salary, deductions;

    public SalarySlip(String name, String code, double basic_pay)
    {
        super(name, code, basic_pay);
        this.deductions=0;
        this.salary=0;
    }
    public void SalaryCalc(){
        deductions=0.5*basic_pay;
        salary=basic_pay-deductions;
    }
    public void print_slip()
    {
        System.out.println("Employee Salary : "+salary);
    }
}
import java.util.Scanner;

public class EmployeeMain {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        SalarySlip employee = null; // Declare it here and initialize
        to null

        System.out.println("Welcome to Employee Management System");

        while (true) {
            System.out.println("Choose an option:");
            System.out.println("1. Create Employee");
            System.out.println("2. Calculate Salary");
            System.out.println("3. Print Salary Slip");
            System.out.println("4. Exit");
            System.out.print("Enter your choice (1, 2, 3, or 4): ");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter employee name: ");
                    String name = scanner.next();
                    System.out.print("Enter employee ID: ");
                    String id = scanner.next();
                    System.out.print("Enter employee basic salary: ");
                    double basicSalary = scanner.nextDouble();

                    employee = new SalarySlip(name, id, basicSalary);
                    System.out.println("Employee created successfully.");
                    break;

                case 2:
                    if (employee == null) {
                        System.out.println("No employee record found.
                        Please create an employee first.");
                    } else {
                        employee.SalaryCalc();
                        System.out.println("Salary calculated
                        successfully.");
                    }
                    break;

                case 3:
                    if (employee == null) {
                        System.out.println("No employee record found.
                        Please create an employee first.");
                    } else {
                        employee.print_slip();
                    }
                    break;

                case 4:
                    System.out.println("Exiting the program.");
                    scanner.close();
                    System.exit(0);

                default:
                    System.out.println("Invalid choice. Please enter
                    1, 2, 3, or 4.");
            }
        }
    }
```

```
}
```

SAMPLE INPUT-OUTPUT



```
Welcome to Employee Management System
Choose an option:
1. Create Employee
2. Calculate Salary
3. Print Salary Slip
4. Exit
Enter your choice (1, 2, 3, or 4): 1
Enter employee name: lana
Enter employee ID: 23443
Enter employee basic salary: 56000
Employee created successfully.
Choose an option:
1. Create Employee
2. Calculate Salary
3. Print Salary Slip
4. Exit
Enter your choice (1, 2, 3, or 4): 2
Salary calculated successfully.
Choose an option:
1. Create Employee
2. Calculate Salary
3. Print Salary Slip
4. Exit
Enter your choice (1, 2, 3, or 4): 3
Employee Salary : 28000.0
Choose an option:
1. Create Employee
2. Calculate Salary
3. Print Salary Slip
4. Exit
Enter your choice (1, 2, 3, or 4): 4
Exiting the program.
```

FINALIZE METHOD

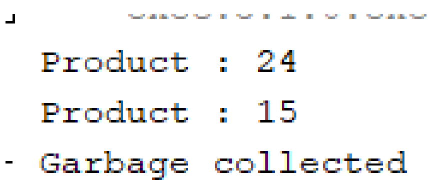
AIM

Write a program to illustrate finalize() method.

PROGRAM

```
package com.mycompany.finalizemethod;

/**
 *
 * @author lanaa
 */
class FinalizeMethod {

    int a,b,c;

    public FinalizeMethod(int a, int b, int c)
    {
        this.a=a;
        this.b=b;
        this.c=c;
    }

    public void printProduct()
    {
        System.out.println("Product : "+(a*b*c));
    }
    public void finalize()
    {
        System.out.println("Garbage collected");
    }
}
public class FinalizeMethodMain {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        FinalizeMethod fm1= new FinalizeMethod(2,3,4);
        FinalizeMethod fm2= new FinalizeMethod(1,3,5);
        fm1.printProduct();
        fm2.printProduct();
        fm1=null;
        System.gc();
    }

}
```

SAMPLE INPUT-OUTPUT



```
Product : 24
Product : 15
Garbage collected
```

TOWER OF HANOI PROBLEM

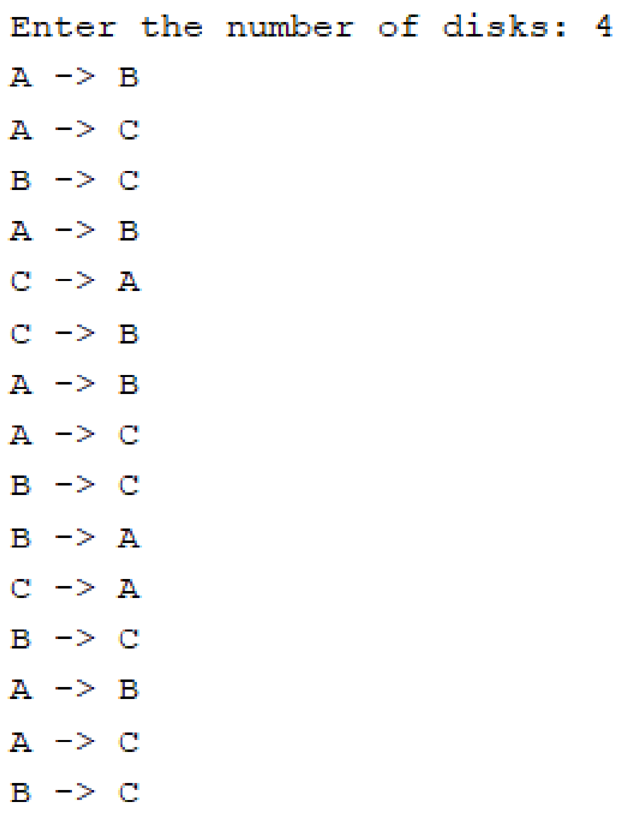
AIM

Write a program to implement the Tower of Hanoi problem using recursion

PROGRAM

```
package LAB_CYCLE_2.Q1;

/**
 *
 * @author lanaa
 */
public class TowerOfHanoi {
    void recursion(int n, char src, char aux, char dest)
    {
        if(n==1)
        {
            System.out.println(src+" -> "+dest);
        }
        else
        {
            recursion(n-1,src,dest,aux);
            recursion(1,src,aux,dest);
            recursion(n-1,aux,src,dest);
        }
    }
}
import java.util.Scanner;

public class TowerOfHanoiMain {

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

        System.out.print("Enter the number of disks: ");
        int numberOfDisks = scanner.nextInt();

        TowerOfHanoi toh = new TowerOfHanoi();
        toh.recursion(numberOfDisks, 'A', 'B', 'C');
        scanner.close();
    }
}
```

SAMPLE INPUT-OUTPUT



```
Enter the number of disks: 4
A -> B
A -> C
B -> C
A -> B
C -> A
C -> B
A -> B
A -> C
B -> C
B -> A
C -> A
B -> C
A -> B
A -> C
B -> C
```

DYNAMIC ARRAY OPERATIONS

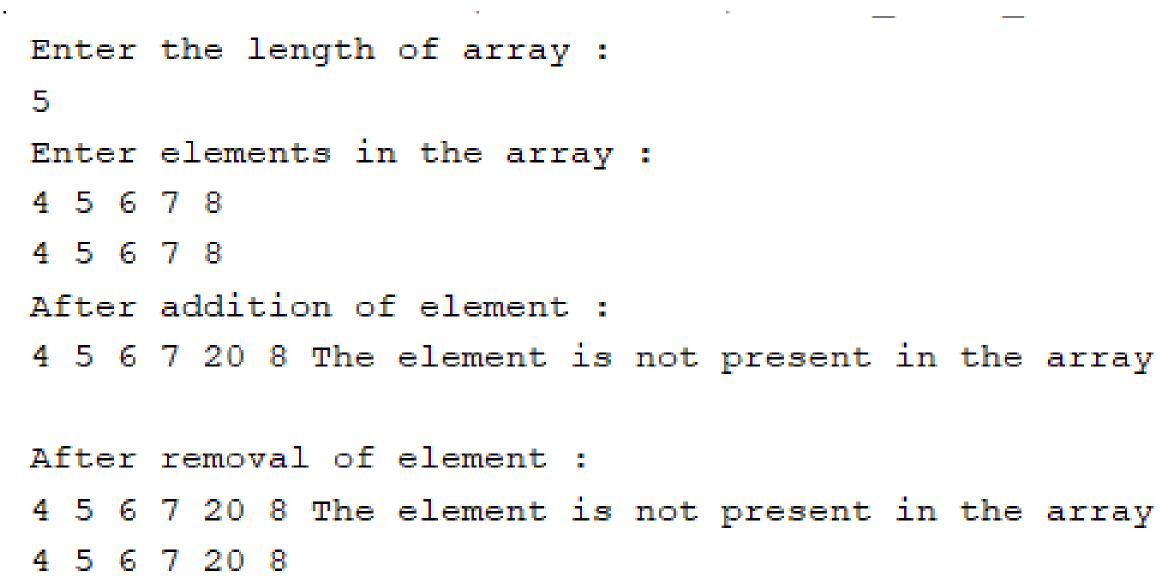
AIM

Write a program to create a class DynamicArray to implement a dynamic array. Provide
a. Constructor to initialize the array
b. Function to print array
c. Function to add elements to a position (if position not specified, add to end)
d. Function to remove elements
e. Function to search an element

PROGRAM

```
import java.util.Scanner;

public class DynamicArray {

    private int[] array;
    private int len;

    public DynamicArray() {
        this.len = 0;
        array = new int[len];
    }

    public DynamicArray(int len) {
        this.len = len;
        array = new int[len];
    }

    public void arrayRead() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter elements in the array : ");
        for (int i = 0; i < len; i++) {
            array[i] = sc.nextInt();
        }
    }

    public void printArray() {
        for (int i = 0; i < len; i++) {
            System.out.print(array[i] + " ");
        }
    }

    public void addElement(int element, int pos) {
        if (pos < 0 || pos > len) {
            System.out.println("Invalid position to add the element.");
            return;
        }

        int[] newArray = new int[len + 1];

        for (int i = 0, j = 0; i < len; i++, j++) {
            if (i == pos) {
                newArray[j] = element;
                j++;
            }
            newArray[j] = array[i];
        }

        array = newArray;
        len++;
    }

    public void removeElement(int element) {
        boolean bool = false;
        for (int i = 0; i < len; i++) {
            if (array[i] == element) {
                for (int j = i; j < len - 1; j++) {
                    array[j] = array[j + 1];
                }
                bool = true;
                break;
            }
        }

        if (bool == true) {
            len--;
        } else {
            System.out.println("The element is not present in the array ");
        }
    }
    public void searchArray(int element) {
        boolean bool = false;
        for (int i = 0; i < len; i++) {
            if (array[i] == element) {
                bool = true;
                break;
            }
        }
        if (bool == true) {
            System.out.println("The element is present in the array ");
        } else {
            System.out.println("The element is not present in the array ");
        }
    }
}

package LAB_CYCLE_2.Q2;

import java.util.Scanner;

/**
 *
 * @author lanaa
 */
public class DynamicArrayMain {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the length of array : ");
        int len=sc.nextInt();
        DynamicArray arr=new DynamicArray(len);
        arr.arrayRead();
        arr.printArray();
        arr.addElement(20, 4);
        System.out.println("\nAfter addition of element : ");
        arr.printArray();
        arr.removeElement(13);
        System.out.println("\nAfter removal of element : ");
        arr.printArray();
        arr.searchArray(24);
        arr.printArray();
    }

}
```

SAMPLE INPUT-OUTPUT



```
Enter the length of array :
5
Enter elements in the array :
4 5 6 7 8
4 5 6 7 8
After addition of element :
4 5 6 7 20 8 The element is not present in the array

After removal of element :
4 5 6 7 20 8 The element is not present in the array
4 5 6 7 20 8
```

PASCAL'S TRIANGLE

AIM

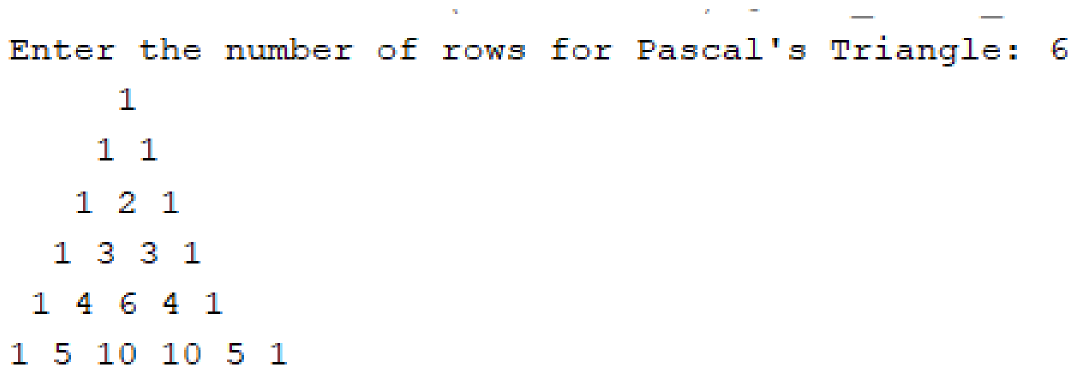
Write a program to Pascal triangle.

PROGRAM

```
package LAB_CYCLE_2.Q3;

/**
 *
 * @author lanaa
 */
class Pascal
{
    public int fact(int n)
    {
        int factorial=1;
        if (n==1 || n==0)
        {
            factorial=1;
        }
        else
        {
            factorial=n*fact(n-1);
        }

        return factorial;
    }

    public void triangle(int m)
    {
        for(int i=0; i<m; i++)
        {
            for (int j=m-i-1; j>0; j--)
            {
                System.out.print(" ");
            }
            for(int k=0; k<i+1; k++)
            {
                System.out.print(fact(i)/(fact(((i)-k))*fact(k))+" ");
            }

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

```
import java.util.Scanner;

public class PascalTriangle {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of rows for Pascal's Triangle: ");
        int numRows = scanner.nextInt();

        Pascal pascalTri = new Pascal();
        pascalTri.triangle(numRows);

        scanner.close();
    }
}
```

SAMPLE INPUT-OUTPUT



```
Enter the number of rows for Pascal's Triangle: 6
     1
    1 1
   1 2 1
  1 3 3 1
 1 4 6 4 1
1 5 10 10 5 1
```

EMPLOYEE LIST

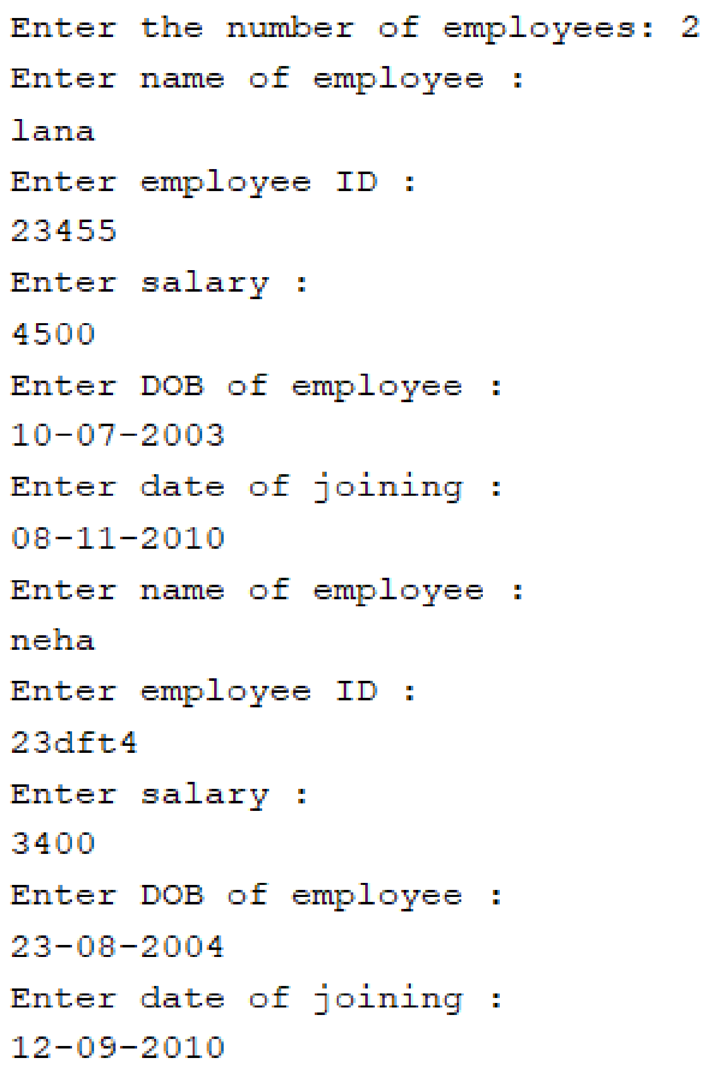
AIM

Write a program to create a class employee having members Employee id, Employee name, date of birth, date of joining, and salary. Read the details of n employees, sort the employee list in the descending order of salary, and print it. (Note use nested class for date of birth and date of joining)

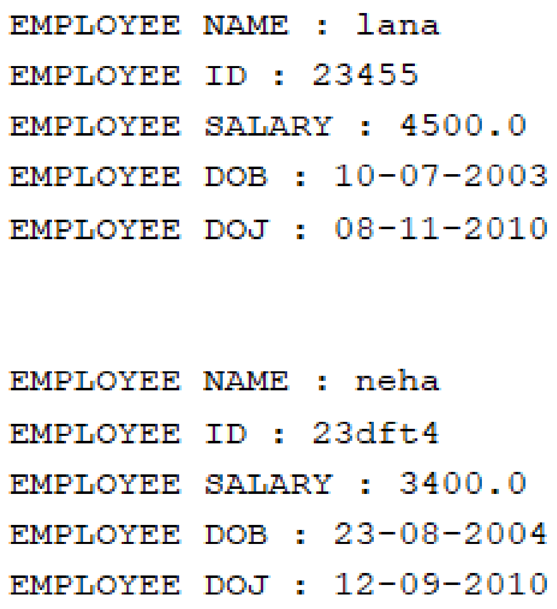
PROGRAM

```
package LAB_CYCLE_2.Q4;

import java.util.Scanner;

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/
 license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/
 Main.java to edit this template
 */

/**
 *
 * @author lanaa
 */
class Emp{

    String EmpId, EmpName;
    double salary;

    public Emp()
    {
        EmpId=EmpName="";
        salary=0;
    }

    public void read()
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter name of employee : ");
        EmpName=sc.nextLine();
        System.out.println("Enter employee ID : ");
        EmpId=sc.nextLine();
        System.out.println("Enter salary : ");
        salary=sc.nextDouble();
    }

    class EmpDate{
        String EmpDOB;
        String EmpDOJ;

        public void read()
        {
            Scanner sc=new Scanner(System.in);
            System.out.println("Enter DOB of employee : ");
            EmpDOB=sc.nextLine();
            System.out.println("Enter date of joining : ");
            EmpDOJ=sc.nextLine();
        }

    }
}

class EmployeeList{
    int count;
    Emp[] employee;
    Emp.EmpDate[] empDate;

    public EmployeeList(int count)
    {
        this.count=count;
        employee=new Emp[count];
        empDate=new Emp.EmpDate[count];
    }

    public void readList()
    {
        for(int i=0; i<count; i++)
        {
            employee[i]=new Emp();
            employee[i].read();
            empDate[i]=employee[i].new EmpDate();
            empDate[i].read();
        }
    }
    public void sortList()
    {
        for(int i=0; i<count; i++)
        {
            Emp temp=employee[i];
            int j=i-1;

            while(j>=0 && temp.salary>employee[j].salary)
            {
                employee[j+1]=employee[j];
                j--;
            }
            employee[j+1]=temp;
        }
    }
    public void display()
    {
        System.out.println("\n");

        for (int k=0; k<count; k++)
        {

            System.out.println("EMPLOYEE NAME : "+employee[k].EmpName);
            System.out.println("EMPLOYEE ID : "+employee[k].EmpId);
            System.out.println("EMPLOYEE SALARY : "+employee[k].salary);
            System.out.println("EMPLOYEE DOB : "+empDate[k].EmpDOB);
            System.out.println("EMPLOYEE DOJ : "+empDate[k].EmpDOJ);

            System.out.println("\n");
        }
    }

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

        System.out.print("Enter the number of employees: ");
        int numEmployees = scanner.nextInt();

        EmployeeList empList = new EmployeeList(numEmployees);
        empList.readList(scanner);
        empList.sortList();
        empList.display();

        scanner.close();
    }
}
```

SAMPLE INPUT-OUTPUT



```
Enter the number of employees: 2
Enter name of employee :
lana
Enter employee ID :
23455
Enter salary :
4500
Enter DOB of employee :
10-07-2003
Enter date of joining :
08-11-2010
Enter name of employee :
neha
Enter employee ID :
23dft4
Enter salary :
3400
Enter DOB of employee :
23-08-2004
Enter date of joining :
12-09-2010
```



```
EMPLOYEE NAME : lana
EMPLOYEE ID : 23455
EMPLOYEE SALARY : 4500.0
EMPLOYEE DOB : 10-07-2003
EMPLOYEE DOJ : 08-11-2010


EMPLOYEE NAME : neha
EMPLOYEE ID : 23dft4
EMPLOYEE SALARY : 3400.0
EMPLOYEE DOB : 23-08-2004
EMPLOYEE DOJ : 12-09-2010
```

CALCULATOR

AIM

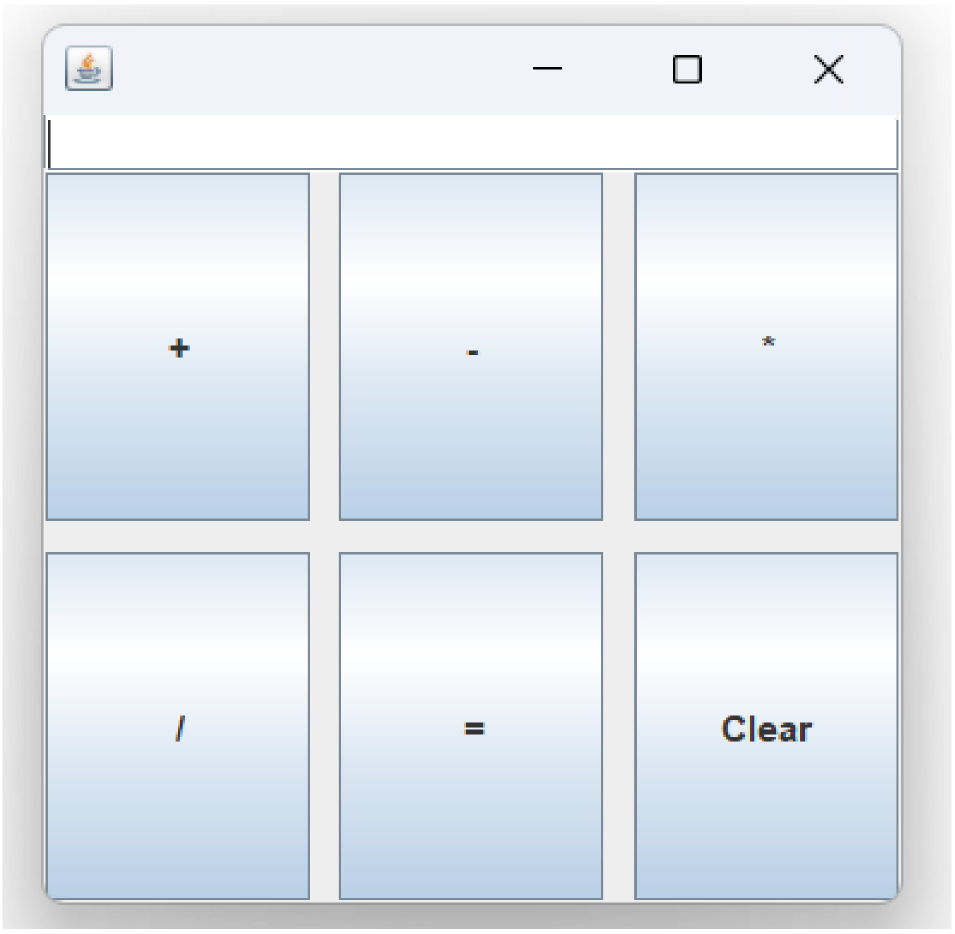
Create a swing program to implement a simple calculator (without drag and drop).

PROGRAM

```
package LAB_CYCLE_2.Q5;

import java.awt.BorderLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class SimpleCalculator extends JFrame {

    JPanel myPanel;
    JButton addBtn, subBtn, equalBtn, clearBtn, multBtn, divBtn;
    JTextField textField1;

    double num1, num2, result;
    char operator;

    public SimpleCalculator() {
        this.setSize(300, 300);
        this.setLocation(900, 500);

        textField1 = new JTextField();

        addBtn = new JButton("+");
        subBtn = new JButton("-");
        equalBtn = new JButton("=");
        multBtn= new JButton("*");
        divBtn= new JButton("/");
        clearBtn = new JButton("Clear");

        myPanel = new JPanel();
        myPanel.setLayout(new GridLayout(2, 2, 10, 10));
        myPanel.add(addBtn);
        myPanel.add(subBtn);
        myPanel.add(multBtn);
        myPanel.add(divBtn);
        myPanel.add(equalBtn);
        myPanel.add(clearBtn);

        this.add(textField1, BorderLayout.NORTH);
        this.add(myPanel, BorderLayout.CENTER);

        addBtn.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                num1 = Double.parseDouble(textField1.getText());
                operator = '+';
                textField1.setText("");
//                textField1.setText("+");
            }
        });

        subBtn.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                num1 = Double.parseDouble(textField1.getText());
                operator = '-';
                textField1.setText("");
//                textField1.setText("-");
            }
        });
        multBtn.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                num1 = Double.parseDouble(textField1.getText());
                operator = '*';
                textField1.setText("");
//                textField1.setText("*");
            }
        });
        divBtn.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                num1 = Double.parseDouble(textField1.getText());
                operator = '/';
                textField1.setText("");
//                textField1.setText("/");
            }
        });

        equalBtn.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                num2 = Double.parseDouble(textField1.getText());
                switch (operator) {
                    case '+':
                        result = num1 + num2;
                        break;
                    case '-':
                        result=num1-num2;
                        break;
                    case '*':
                        result=num1*num2;
                        break;
                    case '/':
                        result=num1/num2;
                        break;

                }

                textField1.setText(String.valueOf(result));
            }
        });

        clearBtn.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                textField1.setText("");
            }
        });

        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setVisible(true);
    }


}
package LAB_CYCLE_2.Q5;
```

```
public class SimpleCalculatorMain {
//
//    /**
//     * @param args the command line arguments
//     */

//    public static void main(String[] args) {
//        SwingUtilities.invokeLater(new Runnable() {
//            public void run() {
//                new SimpleCalculator();
//            }
//        });
//    }
//}
    public static void main(String[] args) {
        new SimpleCalculator();
    }
}
```

SAMPLE INPUT-OUTPUT



EXCEPTION HANDLING

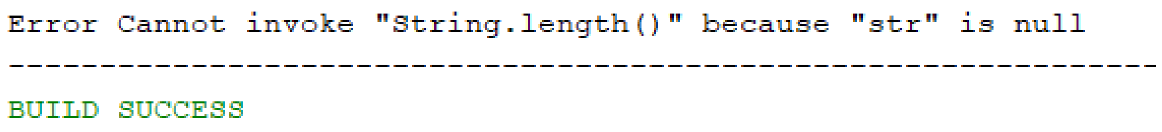
AIM

Write a program to illustrate exception handling in Java for the following exception.
a. Number format exception
b. Null point exception

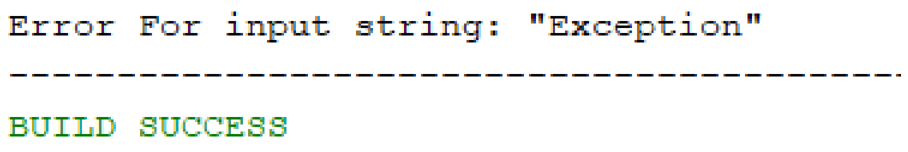
PROGRAM

```
package LAB_CYCLE_2.Q6;

/**
 *
 * @author lanaa
 */
public class NullPointEx {

    /**
     * @param args the command line arguments
     */


    public static void main(String[] args) {
        // TODO code application logic here
        String[] names=new String[3];
        names[0]= "Lana";
        names[1]="Neha";

        String str=names[2];



        try{
            int len=str.length();
            System.out.println(len);
        }
        catch(NullPointerException ex)
        {
            System.out.println("Error "+ex.getMessage());
        }
    }
}
package LAB_CYCLE_2.Q6;

/**
 *
 * @author lanaa
 */
public class NumFormatEx {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        String invalidString= "Exception";

        try{

            int number = Integer.parseInt(invalidString);
            System.out.println("number = "+invalidString);
        }
        catch(NumberFormatException ex)
        {
            System.out.println("Error "+ex.getMessage());
        }
    }

}
```

SAMPLE INPUT-OUTPUT



```
Error Cannot invoke "String.length()" because "str" is null
---------------------------------------------------------------
BUILD SUCCESS
```



```
Error For input string: "Exception"
-----------------------------------------
BUILD SUCCESS
```

EXCEPTION HANDLING - GUI

AIM

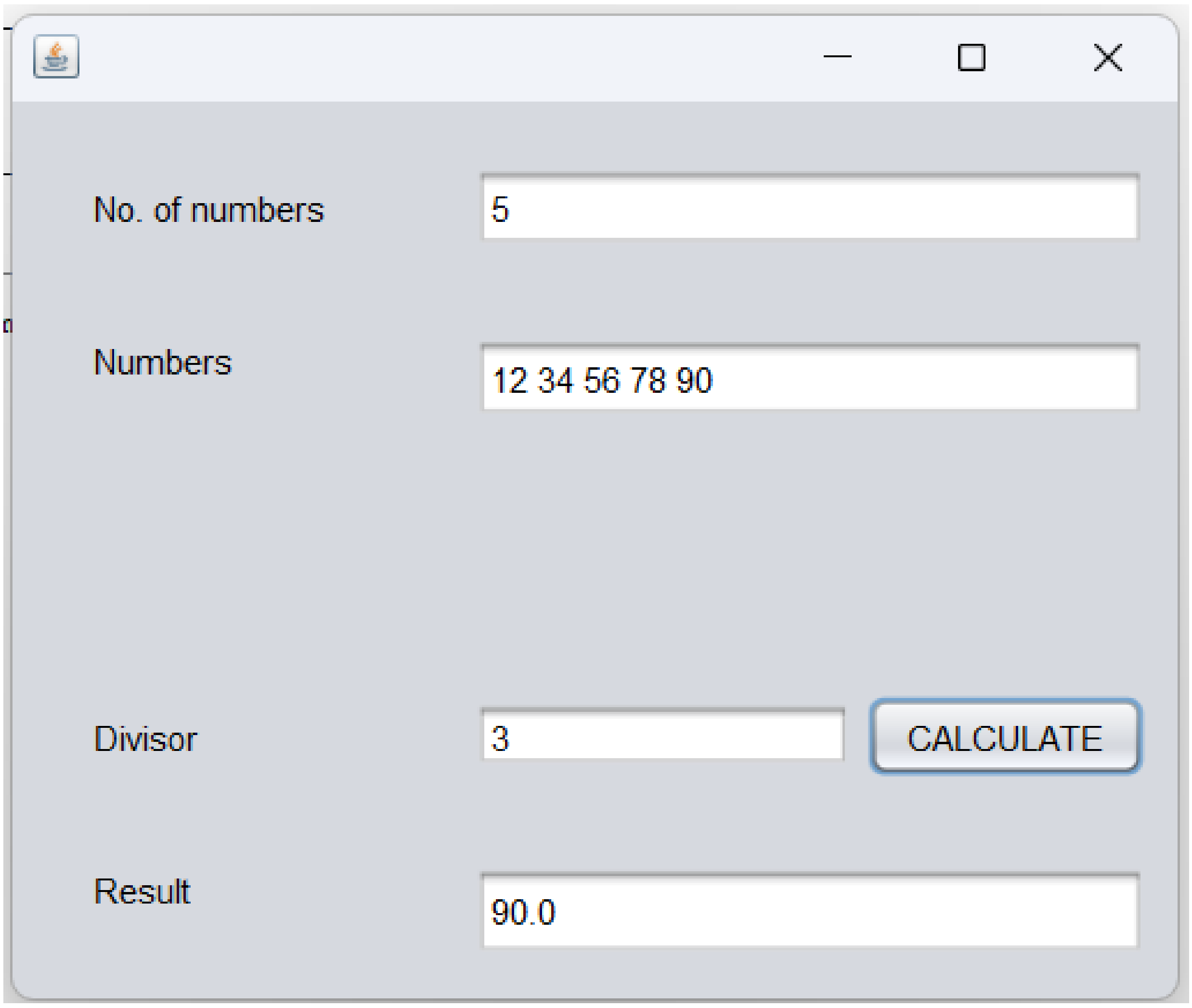
Write a program to create the following GUI:
To read element s to an array, divide the sum of element s with the given divisor and print the result. Handle all exceptions and alert the user using dialogue.

PROGRAM

```
package LAB_CYCLE_2.Q7;

/**
 *
 * @author lanaa
 */
public class NumberGUI extends javax.swing.JFrame {

    /**
     * Creates new form NumberGUI
     */
    public NumberGUI() {
        initComponents();
    }

    /**
     * This method is called from within the constructor to
     initialize the form.
     * WARNING: Do NOT modify this code. The content of this
     method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    //GEN-BEGIN:initComponents
    private void initComponents() {

        count = new javax.swing.JTextField();
        jLabel1 = new javax.swing.JLabel();
        array = new javax.swing.JTextField();
        jLabel2 = new javax.swing.JLabel();
        div = new javax.swing.JTextField();
        calc_btn = new javax.swing.JButton();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        result = new javax.swing.JTextField();

        setDefaultCloseOperation(javax.swing.WindowConstants.
        EXIT_ON_CLOSE);

        count.addActionListener(new java.awt.event.ActionListener()
        {
            public void actionPerformed(java.awt.event
            .ActionEvent evt) {
                countActionPerformed(evt);
            }
        });

        jLabel1.setText("No. of numbers");

        array.addActionListener(new java.awt.event
        .ActionListener() {
            public void actionPerformed(java.awt.event.
            ActionEvent evt) {
                arrayActionPerformed(evt);
            }
        });

        jLabel2.setText("Numbers");

        div.addActionListener(new java.awt.event
        .ActionListener() {
            public void actionPerformed(java.awt
            .event.ActionEvent evt) {
                divActionPerformed(evt);
            }
        });

        calc_btn.setText("CALCULATE");
        calc_btn.addActionListener(new java.awt.event
        .ActionListener() {
            public void actionPerformed(java.awt.event
            .ActionEvent evt) {
                calc_btnActionPerformed(evt);
            }
        });

        jLabel3.setText("Divisor");

        jLabel4.setText("Result");

        result.addActionListener(new java.awt.event
        .ActionListener() {
            public void actionPerformed(java.awt.event
            .ActionEvent evt) {
                resultActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing
        .GroupLayout
        (getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing
            .GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(28, 28, 28)
                .addGroup(layout.createParallelGroup
                (javax.swing.GroupLayout.Alignment
                .LEADING)
                    .addGroup(layout.createParallelGroup
                    (javax.swing.GroupLayout.
                    Alignment.LEADING, false)
                        .addComponent(jLabel1, javax.swing
                        .GroupLayout.DEFAULT_SIZE,
                        114, Short.MAX_VALUE)
                        .addComponent(jLabel2, javax.swing
                        .GroupLayout.DEFAULT_SIZE, javax
                        .swing.GroupLayout.
                        DEFAULT_SIZE, Short.MAX_VALUE))
                    .addComponent(jLabel3, javax.swing.
                    GroupLayout.PREFERRED_SIZE
                    , 103, javax.swing.GroupLayout
                    .PREFERRED_SIZE)
                    .addComponent(jLabel4, javax.swing.
                    GroupLayout.PREFERRED_SIZE, 103,
                    javax.swing.GroupLayout.
                    PREFERRED_SIZE))
                .addGap(18, 18, 18)
                .addGroup(layout.createParallelGroup
                (javax.swing.GroupLayout.Alignment
                .LEADING, false)
                    .addComponent(array)
                    .addComponent(count)
                    .addGroup(javax.swing.GroupLayout
                    .Alignment.TRAILING, layout.
                    createSequentialGroup()
                        .addComponent(div, javax.swing.
                        GroupLayout.PREFERRED_SIZE,
                        130, javax.swing.GroupLayout
                        .PREFERRED_SIZE)
                        .addPreferredGap(javax.swing.
                        LayoutStyle.ComponentPlacement
                        .RELATED)
                        .addComponent(calc_btn))
                    .addComponent(result))
                .addContainerGap(12, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.
            GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(23, 23, 23)
                .addGroup(layout.createParallelGroup
                (javax.swing.GroupLayout.Alignment
                .BASELINE)
                    .addComponent(count, javax.swing.
                    GroupLayout.PREFERRED_SIZE, 28,
                    javax.swing.GroupLayout.
                    PREFERRED_SIZE)
                    .addComponent(jLabel1))
                .addGap(31, 31, 31)
                .addGroup(layout.createParallelGroup
                (javax.swing.GroupLayout.
                Alignment.LEADING)
                    .addComponent(jLabel2)
                    .addComponent(array, javax.swing.
                    GroupLayout.PREFERRED_SIZE, javax
                    .swing.GroupLayout.DEFAULT_SIZE,
                    javax.swing.GroupLayout.
                    PREFERRED_SIZE))
                .addGap(96, 96, 96)
                .addGroup(layout.createParallelGroup(javax.
                swing.GroupLayout.
                Alignment.BASELINE)
                    .addComponent(div, javax.swing
                    .GroupLayout.PREFERRED_SIZE, 23,
                    javax.swing.GroupLayout.
                    PREFERRED_SIZE)
                    .addComponent(calc_btn)
                    .addComponent(jLabel3))
                .addPreferredGap(javax.swing.LayoutStyle.
                ComponentPlacement.RELATED,
                31, Short.MAX_VALUE)
                .addGroup(layout.createParallelGroup
                (javax.swing.GroupLayout.
                Alignment.LEADING)
                    .addComponent(jLabel4)
                    .addComponent(result,
                    javax.swing.GroupLayout
                    .PREFERRED_SIZE, 31, javax.swing
                    .GroupLayout.PREFERRED_SIZE))
                .addGap(15, 15, 15))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void arrayActionPerformed(java.awt.event
    .ActionEvent evt) {//GEN-FIRST:event_arrayActionPerformed
        // TODO add your handling code here:


    }//GEN-LAST:event_arrayActionPerformed
    private void divActionPerformed(java.awt.event
    .ActionEvent evt) {//GEN-FIRST:event_divActionPerformed
        // TODO add your handling code here:
    }//GEN-LAST:event_divActionPerformed

    private void resultActionPerformed(java.awt.event
    .ActionEvent evt) {//GEN-FIRST:event_resultActionPerformed
        // TODO add your handling code here:

    }//GEN-LAST:event_resultActionPerformed

    private void countActionPerformed(java.awt.event
    .ActionEvent evt) {//GEN-FIRST:event_countActionPerformed
        // TODO add your handling code here:


    }//GEN-LAST:event_countActionPerformed

    private void calc_btnActionPerformed(java.awt.event
    .ActionEvent evt) {//GEN-FIRST:event_calc_btnActionPerformed
        // TODO add your handling code here:
        try {
            int count_no = Integer.parseInt(count.getText());

            String[] inputValues = array.getText().split("\\s+");

            double[] num = new double[count_no];

            for (int i = 0; i < count_no; i++) {
                num[i] = Double.parseDouble(inputValues[i]);
            }

            double divisor = Double.parseDouble(div.getText());

            double sum = 0;

            for (int i = 0; i < count_no; i++) {
                sum += num[i];
            }

            double quotient = sum / divisor;
            calc_btn.isSelected();
            result.setText(Double.toString(quotient));
        } catch (NumberFormatException ex) {
            // Handle NumberFormatException
            result.setText("Invalid input. Please
            enter valid integers.");
        } catch (ArrayIndexOutOfBoundsException ex1)
        {
            // Handle ArrayIndexOutOfBoundsException
            result.setText("Array index out of bounds
            . Please check your input.");
        }
    }//GEN-LAST:event_calc_btnActionPerformed

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc
        =" Look and feel
        setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not
        available, stay with the
        default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/
         uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
            javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel
                    (info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger
            (NumberGUI.class.getName())
```

```
                .log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger
            (NumberGUI.class.getName())
            .log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger
            (NumberGUI.class.getName())
            .log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger
            (NumberGUI.class.getName())
            .log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new NumberGUI().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JTextField array;
    private javax.swing.JButton calc_btn;
    private javax.swing.JTextField count;
    private javax.swing.JTextField div;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JTextField result;
    // End of variables declaration//GEN-END:variables
}
```

SAMPLE INPUT-OUTPUT



No. of numbers: 5

Numbers: 12 34 56 78 90

Divisor: 3

Result: 90.0

USER DEFINED EXCEPTIONS

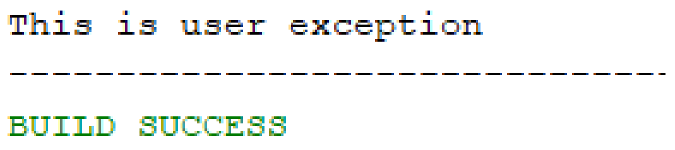
AIM

Write a program to illustrate user-defined exceptions.

PROGRAM

```
package LAB_CYCLE_2.Q8;

/**
 *
 * @author lanaa
 */
public class UserException extends Exception{

    /**
     * @param args the command line arguments
     */
    public UserException(String str)
    {
        super(str);
    }
    public static void main(String[] args) {
        // TODO code application logic here
        try{
            throw new UserException("This is user exception ");
        }
        catch(UserException ex)
        {
            System.out.println(ex.getMessage());
        }
    }

}
```

SAMPLE INPUT-OUTPUT



```
This is user exception
------------------------------------
BUILD SUCCESS
```

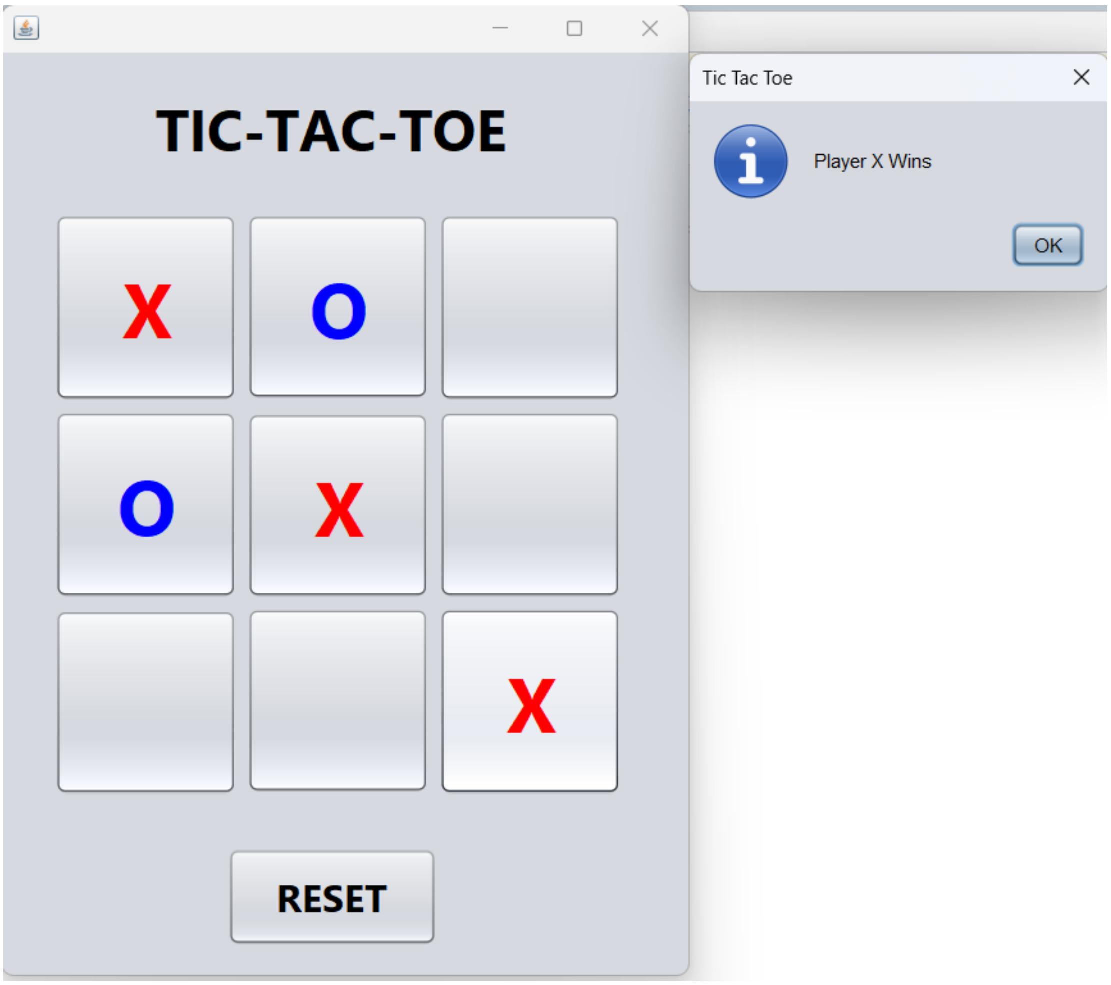
TIC-TAC-TOE

AIM

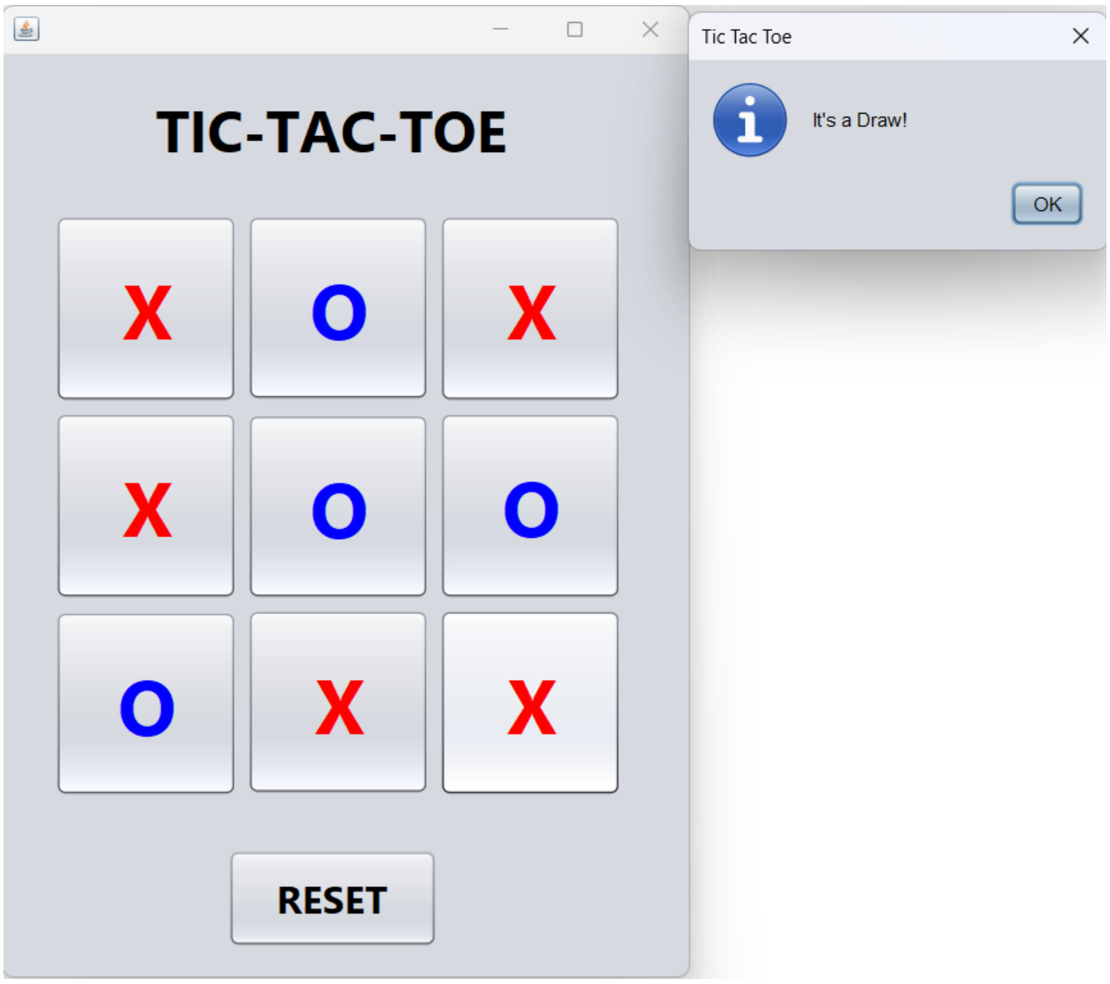
Write a program to create a 2-player tic-tac-toe game (using a grid layout)flushleft
PROGRAM

```
package LAB_CYCLE_2.Q9;

import java.awt.Color;
import java.awt.Component;
import javax.swing.JFrame;
import javax.swing.JOptionPane;

/**
 *
 * @author lanaa
 */
public class TicTacToe extends javax.swing.JFrame {
    private JFrame frame;
    private int xCount = 0;
    private int oCount = 0;
    private String startGame = "X";
    private int b1 = 10;
    private int b2 = 10;
    private int b3 = 10;
    private int b4 = 10;
    private int b5 = 10;
    private int b6 = 10;
    private int b7 = 10;
    private int b8 = 10;
    private int b9 = 10;

    /**
     * Creates new form TicTacToe
     */
    public TicTacToe() {
        initComponents();
    }

    private void winningGame() {
        if (b1 == 1 && b2 == 1 && b3 == 1) {
            JOptionPane.showMessageDialog
            (frame, "Player X Wins", "Tic Tac Toe",

            JOptionPane.INFORMATION_MESSAGE);

        } else if (b4 == 1 && b5 == 1 && b6 == 1) {

            JOptionPane.showMessageDialog
            (frame, "Player X Wins", "Tic Tac Toe",

            JOptionPane.INFORMATION_MESSAGE);

        } else if (b7 == 1 && b8 == 1 && b9 == 1) {

            JOptionPane.showMessageDialog
            (frame, "Player X Wins", "Tic Tac Toe",

            JOptionPane.INFORMATION_MESSAGE);

        } else if (b1 == 1 && b4 == 1 && b7 == 1) {

            JOptionPane.showMessageDialog
            (frame, "Player X Wins", "Tic Tac Toe",

            JOptionPane.INFORMATION_MESSAGE);

        } else if (b2 == 1 && b5 == 1 && b8 == 1) {

            JOptionPane.showMessageDialog
            (frame, "Player X Wins", "Tic Tac Toe",

            JOptionPane.INFORMATION_MESSAGE);

        } else if (b3 == 1 && b6 == 1 && b9 == 1) {

            JOptionPane.showMessageDialog
            (frame, "Player X Wins", "Tic Tac Toe",

            JOptionPane.INFORMATION_MESSAGE);

        } else if (b1 == 1 && b5 == 1 && b9 == 1) {
            JOptionPane.showMessageDialog
            (frame, "Player X Wins", "Tic Tac Toe",

            JOptionPane.INFORMATION_MESSAGE);

        } else if (b3 == 1 && b5 == 1 && b7 == 1) {
            JOptionPane.showMessageDialog
            (frame, "Player X Wins", "Tic Tac Toe",
            JOptionPane.INFORMATION_MESSAGE);

        } else if (b1 != 10 && b2 != 10 && b3 != 10
                && b4 != 10 && b5 != 10 && b6 != 10
                && b7 != 10 && b8 != 10 && b9 != 10) {
            JOptionPane.showMessageDialog
            (frame, "It’s a Draw!", "Tic Tac Toe",
            JOptionPane.INFORMATION_MESSAGE);
        }
    }

    private void changePlayer() {
        if (startGame.equalsIgnoreCase("X")) {
            startGame = "O";
        } else {
            startGame = "X";
        }
    }

    /**
     * This method is called from within
     the constructor to initialize the form.
     * WARNING: Do NOT modify this code.
     The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
    // Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jPanel2 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        btn1 = new javax.swing.JButton();
        btn2 = new javax.swing.JButton();
        btn4 = new javax.swing.JButton();
        btn5 = new javax.swing.JButton();
        btn6 = new javax.swing.JButton();
        btn3 = new javax.swing.JButton();
        btn7 = new javax.swing.JButton();
        btn8 = new javax.swing.JButton();
        btn9 = new javax.swing.JButton();
        resetbtn = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing
        .WindowConstants.EXIT_ON_CLOSE);

        jLabel1.setFont(new java.awt.
        Font("Segoe UI", 1, 36)); // NOI18N
        jLabel1.setHorizontalAlignment
        (javax.swing.SwingConstants.CENTER);
        jLabel1.setText("TIC-TAC-TOE");

        javax.swing.GroupLayout jPanel2Layout
        = new javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(
                jPanel2Layout.createParallelGroup
                (javax.swing.GroupLayout.Alignment.
                LEADING)
                        .addGroup(jPanel2Layout.
                        createSequentialGroup()
                                .addGap(90, 90, 90)
                                .addComponent
                                (jLabel1, javax.swing.
                                GroupLayout.DEFAULT_SIZE,
                                        javax.swing.
                                        GroupLayout.
                                        DEFAULT_SIZE,
                                        Short.MAX_VALUE)
                                .addGap(293, 293, 293)));
        jPanel2Layout.setVerticalGroup(
                jPanel2Layout.createParallelGroup
                (javax.swing.GroupLayout.Alignment.
                LEADING)
                        .addGroup(javax.swing.
                        GroupLayout.Alignment.TRAILING,
                        jPanel2Layout.createSequentialGroup()
                                .addGap(16, 16, 16)
                                .addComponent(jLabel1,
                                javax.swing.GroupLayout
                                .DEFAULT_SIZE, 63, Short.MAX_VALUE)
                                .addContainerGap()));

        btn1.setFont(new java.awt.Font("Segoe UI", 1, 48));
        // NOI18N
        btn1.addActionListener(new java.awt.event
        .ActionListener() {
            public void actionPerformed(java.awt
            .event.ActionEvent evt) {
                btn1ActionPerformed(evt);
            }
        });

        btn2.setFont(new java.awt.Font("Segoe UI", 1, 48));
        // NOI18N
        btn2.addActionListener(new java.awt.event
        .ActionListener() {
            public void actionPerformed(java.awt
            .event.ActionEvent evt) {
                btn2ActionPerformed(evt);
            }
        });

        btn4.setFont(new java.awt.Font("Segoe UI", 1, 48));
        // NOI18N
        btn4.addActionListener(new java.awt.event
        .ActionListener() {
            public void actionPerformed(java.awt
            .event.ActionEvent evt) {
                btn4ActionPerformed(evt);
            }
        });
        btn5.setFont(new java.awt.Font("Segoe UI", 1, 48));
        // NOI18N
        btn5.addActionListener(new java.awt.
        event.ActionListener() {
            public void actionPerformed(java.awt.event.
            ActionEvent evt) {
                btn5ActionPerformed(evt);
            }
        });

        btn6.setFont(new java.awt.Font("Segoe UI", 1, 48)); // NOI18N
        btn6.addActionListener(new java.awt.event.ActionListener()
            public void actionPerformed(java.awt.event.ActionEvent evt)
                    btn6ActionPerformed(evt);
            }
        });

        btn3.setFont(new java.awt.Font("Segoe UI", 1, 48)); // NOI18N
        btn3.addActionListener(new java.awt.event.ActionListener()
            public void actionPerformed(java.awt.event.ActionEvent evt)
                    btn3ActionPerformed(evt);
            }
        });

        btn7.setFont(new java.awt.Font("Segoe UI", 1, 48)); // NOI18N
        btn7.addActionListener(new java.awt.event.ActionListener()
        {public void actionPerformed(java.awt.event.ActionEvent evt)
            {
btn7ActionPerformed(evt);
            }
        });

        btn8.setFont(new java.awt.Font("Segoe UI", 1, 48)); // NOI18N
        btn8.addActionListener(new java.awt.event.ActionListener()
        {public void actionPerformed(java.awt.event.ActionEvent evt)
            {
btn8ActionPerformed(evt);
            }
        });
        btn9.setFont(new java.awt.Font("Segoe UI", 1, 48)); // NOI18N
        btn9.addActionListener(new java.awt.event.ActionListener()
        {public void actionPerformed(java.awt.event.ActionEvent evt)
            {
btn9ActionPerformed(evt);
            }
        });

javax.swing.GroupLayout layout = new javax.swing.GroupLayout
(getContentPane());
getContentPane().setLayout(layout);

    private void btn7ActionPerformed(java.awt.event.ActionEvent evt)
    {// GEN-FIRST:event_btn7ActionPerformed
        // TODO add your handling code here:
        btn7.setText(startGame);
        if (startGame.equalsIgnoreCase("X")) {
            btn7.setForeground(Color.red);
            b7 = 1;
        } else {
            btn7.setForeground(Color.blue);
            b7 = 0;
        }
        changePlayer();
        winningGame();
    }// GEN-LAST:event_btn7ActionPerformed

    private void btn5ActionPerformed(java.awt.event.ActionEvent evt)
    {// GEN-FIRST:event_btn5ActionPerformed
        // TODO add your handling code here:
        btn5.setText(startGame);
        if (startGame.equalsIgnoreCase("X")) {
            btn5.setForeground(Color.red);
            b5 = 1;
        } else {
            btn5.setForeground(Color.blue);
            b5 = 0;
        }
        changePlayer();
        winningGame();
    }// GEN-LAST:event_btn5ActionPerformed
    private void btn6ActionPerformed(java.awt.event.ActionEvent evt)
    {// GEN-FIRST:event_btn6ActionPerformed
        // TODO add your handling code here:
        btn6.setText(startGame);
        if (startGame.equalsIgnoreCase("X")) {
            btn6.setForeground(Color.red);
            b6 = 1;
        } else {
            btn6.setForeground(Color.blue);
            b6 = 0;
        }
        changePlayer();
        winningGame();
    }// GEN-LAST:event_btn6ActionPerformed

    private void btn9ActionPerformed(java.awt.event.ActionEvent evt)
    {// GEN-FIRST:event_btn9ActionPerformed
        // TODO add your handling code here:
        btn9.setText(startGame);
        if (startGame.equalsIgnoreCase("X")) {
            btn9.setForeground(Color.red);
            b9 = 1;
        } else {
            btn9.setForeground(Color.blue);
            b9 = 0;
        }
        changePlayer();
        winningGame();
    }// GEN-LAST:event_btn9ActionPerformed

    private void btn1ActionPerformed(java.awt.event.ActionEvent evt)
    {// GEN-FIRST:event_btn1ActionPerformed
        // TODO add your handling code here:
        btn1.setText(startGame);
        if (startGame.equalsIgnoreCase("X")) {
            btn1.setForeground(Color.red);
            b1 = 1;

        } else {
            btn1.setForeground(Color.blue);
            b1 = 0;
        }
        changePlayer();
        winningGame();
    }// GEN-LAST:event_btn1ActionPerformed

    private void btn2ActionPerformed(java.awt.event.ActionEvent evt)
    {// GEN-FIRST:event_btn2ActionPerformed
        // TODO add your handling code here:
        btn2.setText(startGame);
        if (startGame.equalsIgnoreCase("X")) {
            btn2.setForeground(Color.red);
            b2 = 1;
        } else {
            btn2.setForeground(Color.blue);
            b2 = 0;
        }
        changePlayer();
    }// GEN-LAST:event_btn2ActionPerformed

    private void btn3ActionPerformed(java.awt.event.ActionEvent evt)
    {// GEN-FIRST:event_btn3ActionPerformed
        // TODO add your handling code here:
        btn3.setText(startGame);
        if (startGame.equalsIgnoreCase("X")) {
            btn3.setForeground(Color.red);
            b3 = 1;
        } else {
            btn3.setForeground(Color.blue);
            b3 = 0;
        }
        changePlayer();
        winningGame();
    }// GEN-LAST:event_btn3ActionPerformed

    private void btn8ActionPerformed(java.awt.event.ActionEvent evt)
    {// GEN-FIRST:event_btn8ActionPerformed
        // TODO add your handling code here:
        btn8.setText(startGame);
        if (startGame.equalsIgnoreCase("X")) {
            btn8.setForeground(Color.red);
            b8 = 1;
        } else {
            btn8.setForeground(Color.blue);
            b8 = 0;
        }
        changePlayer();
        winningGame();
    }// GEN-LAST:event_btn8ActionPerformed

    private void btn4ActionPerformed(java.awt.event.ActionEvent evt)
    {// GEN-FIRST:event_btn4ActionPerformed
        // TODO add your handling code here:
        btn4.setText(startGame);
        if (startGame.equalsIgnoreCase("X")) {
            btn4.setForeground(Color.red);
            b4 = 1;
        } else {
            btn4.setForeground(Color.blue);
            b4 = 0;
        }
        changePlayer();
        winningGame();
    }// GEN-LAST:event_btn4ActionPerformed

    private void resetbtnActionPerformed(java.awt.event.ActionEvent evt)
    {// GEN-FIRST:event_resetbtnActionPerformed
        // TODO add your handling code here:
        btn1.setText(null);
        btn2.setText(null);
        btn3.setText(null);
        btn4.setText(null);
        btn5.setText(null);
        btn6.setText(null);
        btn7.setText(null);
        btn8.setText(null);
        btn9.setText(null);
    }// GEN-LAST:event_resetbtnActionPerformed

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        // <editor-fold defaultstate="collapsed" desc=" Look and feel
        setting code
        // (optional) ">
        /*
         * If Nimbus (introduced in Java SE 6) is not available, stay
         with the default
         * look and feel.
         * For details see
         * http://download.oracle.com/javase/tutorial/uiswing
         /lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
            javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.
                    getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(TicTacToe.class.
            getName())
            .log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(TicTacToe.class.
            getName())
            .log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(TicTacToe.class.
            getName())
            .log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(TicTacToe.class.
            getName())
            .log(java.util.logging.Level.SEVERE, null, ex);
        }
        // </editor-fold>
        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new TicTacToe().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton btn1;
    private javax.swing.JButton btn2;
    private javax.swing.JButton btn3;
    private javax.swing.JButton btn4;
    private javax.swing.JButton btn5;
    private javax.swing.JButton btn6;
    private javax.swing.JButton btn7;
    private javax.swing.JButton btn8;
    private javax.swing.JButton btn9;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JButton resetbtn;
    // End of variables declaration//GEN-END:variables
}
```

SAMPLE INPUT-OUTPUT



Player X Wins



It's a Draw!

TREE SET COLLECTION

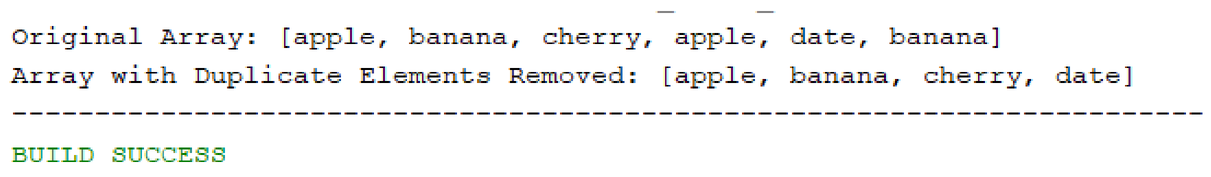
AIM

Write a program to remove duplicate elements from a string array using tree set collection

PROGRAM

```
import java.util.Arrays;
import java.util.TreeSet;

public class RemoveDuplicatesFromArray {
    public static void main(String[] args) {
        String[] stringArray = {"apple", "banana", "cherry",
        "apple", "date", "banana"};

        // Create a TreeSet to store unique elements
        TreeSet<String> uniqueElements = new TreeSet<>();

        // Iterate through the string array and add elements to the TreeSet
        for (String element : stringArray) {
            uniqueElements.add(element);
        }

        // Convert the TreeSet back to an array
        String[] uniqueArray = uniqueElements.toArray(new String[0]);

        // Print the unique elements
        System.out.println("Original Array: " +
        Arrays.toString(stringArray));
        System.out.println("Array with Duplicate Elements Removed: " +
        Arrays.toString(uniqueArray));
    }
}
```

SAMPLE INPUT-OUTPUT



```
Original Array: [apple, banana, cherry, apple, date, banana]
Array with Duplicate Elements Removed: [apple, banana, cherry, date]
------------------------------------------------------------------------
BUILD SUCCESS
```

VALIDATION FORM

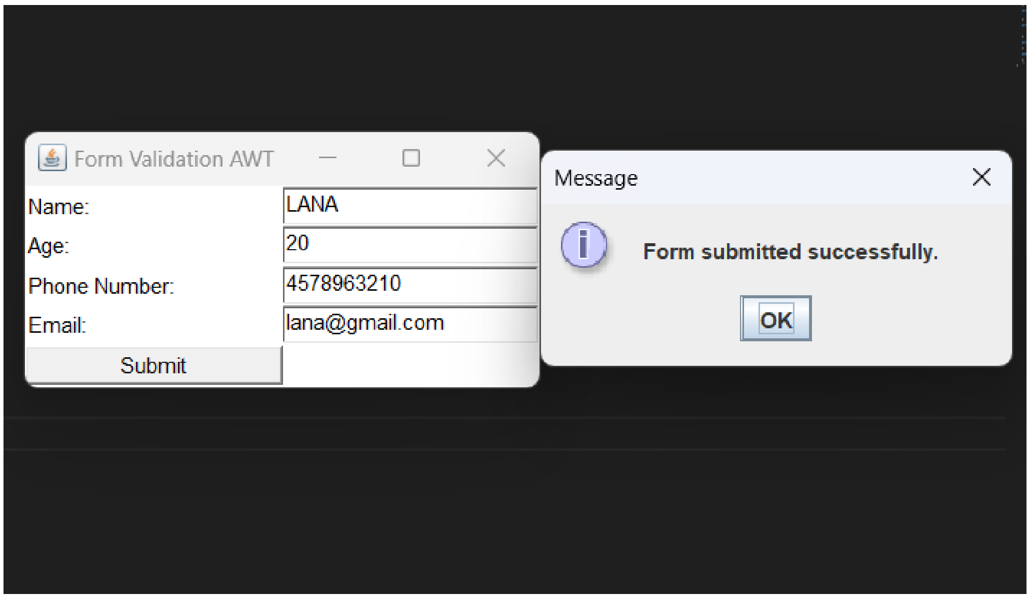
AIM

Write an AWT program for validating the form having a numeric field, character field, phone number, and email id.

PROGRAM

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

public class FormValidationAWT extends Frame implements ActionListener {
    Label nameLabel, ageLabel, phoneLabel, emailLabel;
    TextField nameField, ageField, phoneField, emailField;
    Button submitButton;

    public FormValidationAWT() {
        setLayout(new GridLayout(5, 2));
        nameLabel = new Label("Name:");
        ageLabel = new Label("Age:");
        phoneLabel = new Label("Phone Number:");
        emailLabel = new Label("Email:");

        nameField = new TextField();
        ageField = new TextField();
        phoneField = new TextField();
        emailField = new TextField();

        submitButton = new Button("Submit");
        submitButton.addActionListener(this);

        add(nameLabel);
        add(nameField);
        add(ageLabel);
        add(ageField);
        add(phoneLabel);
        add(phoneField);
        add(emailLabel);
        add(emailField);
        add(submitButton);
        setTitle("Form Validation AWT");
        setSize(300, 150);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == submitButton) {
            String name = nameField.getText();
            String age = ageField.getText();
            String phone = phoneField.getText();
            String email = emailField.getText();

            if (name.isEmpty() || !name.matches("^[a-zA-Z\\s]*$")) {
                showMessage("Invalid name. Please enter a valid name.");
            } else if (age.isEmpty() || !age.matches("^[0-9]+$")) {
                showMessage("Invalid age. Please enter a valid
                numeric age.");
            } else if (phone.isEmpty() || !isValidPhoneNumber(phone)) {
                showMessage("Invalid phone number.
                Please enter a valid phone number.");
            } else if (email.isEmpty() || !isValidEmail(email)) {
                showMessage("Invalid email.
                Please enter a valid email address.");
            } else {
                showMessage("Form submitted successfully.");
                System.exit(0);
            }
        }
    }

    private boolean isValidPhoneNumber(String phoneNumber) {
        // Basic phone number validation (10-digit)
        return phoneNumber.matches("^[0-9]{10}$");
    }

    private boolean isValidEmail(String email) {
        // Email validation using a simple regular expression
        String emailPattern =
        "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,4}$";
        return email.matches(emailPattern);
    }

    private void showMessage(String message) {
        JOptionPane.showMessageDialog(this, message);
    }

    public static void main(String[] args) {
        new FormValidationAWT();
    }
    // System.exit(0);
}
```

SAMPLE INPUT-OUTPUT



```
Form Validation AWT
Name:           LANA
Age:            20
Phone Number:   4578963210
Email:          lana@gmail.com
Submit

Message
Form submitted successfully.
OK
```

WINDOWS 95 APPLICATIONS

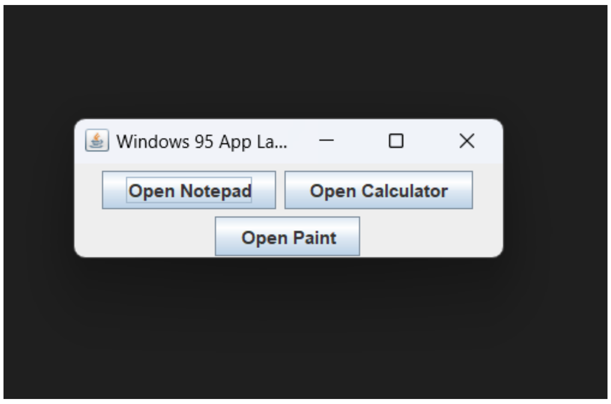
AIM

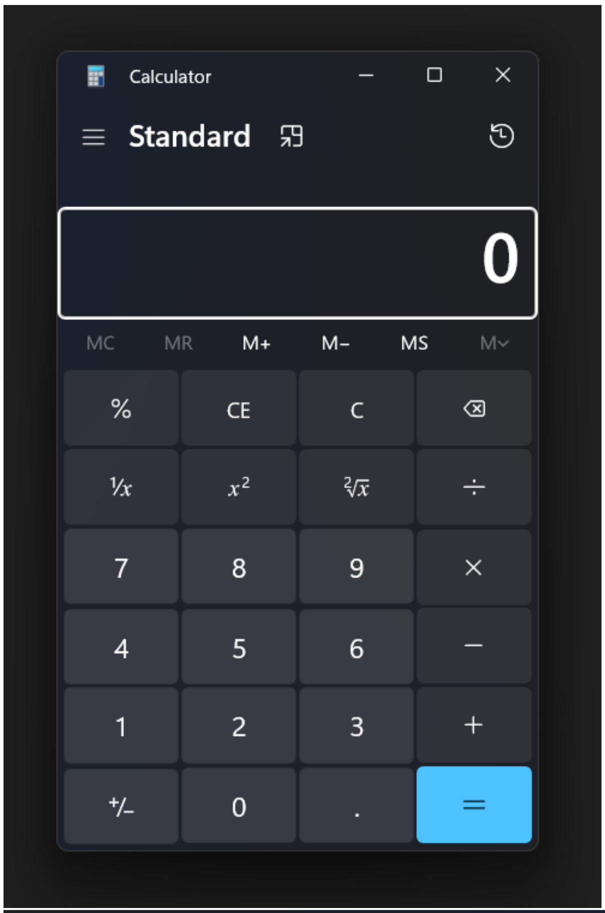
Write a GUI program to execute 3 Windows 95 applications (Like notepad, calculator, paint) through Java

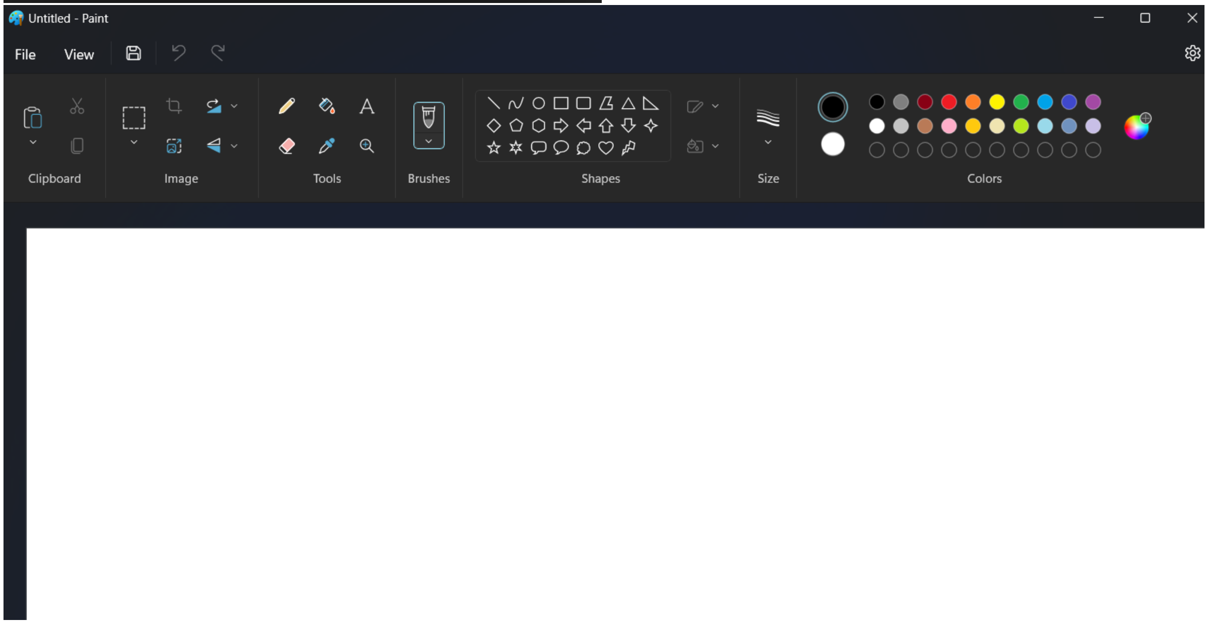
PROGRAM

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

public class Win95AppLauncher extends JFrame {
    public Win95AppLauncher() {
        setTitle("Windows 95 App Launcher");
        setSize(300, 100);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JButton notepadButton = new JButton("Open Notepad");
        JButton calculatorButton = new JButton("Open Calculator");
        JButton paintButton = new JButton("Open Paint");

        notepadButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try {
                    Runtime.getRuntime().exec("notepad");
                } catch (Exception ex) {
                    ex.printStackTrace();
                }
            }
        });

        calculatorButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try {
                    Runtime.getRuntime().exec("calc");
                } catch (Exception ex) {
                    ex.printStackTrace();
                }
            }
        });
        paintButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try {
                    Runtime.getRuntime().exec("mspaint");
                } catch (Exception ex) {
                    ex.printStackTrace();
                }
            }
        });

        JPanel panel = new JPanel();
        panel.add(notepadButton);
        panel.add(calculatorButton);
        panel.add(paintButton);

        add(panel);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                Win95AppLauncher launcher = new Win95AppLauncher();
                launcher.setVisible(true);
            }
        });
    }
}
```

SAMPLE INPUT-OUTPUT




Calculator
Standard
0
MC
MR
M+
M−
MS
M⌄
%
CE
C
⅟x
x^2
$\sqrt[2]{x}$
÷
7
8
9
×
4
5
6
−
1
2
3
+
+/−
0
.
=


Untitled - Paint
File
View
Clipboard
Image
Tools
Brushes
Shapes
Size
Colors

Untitled
File
Edit
View
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8

GARBAGE COLLECTION

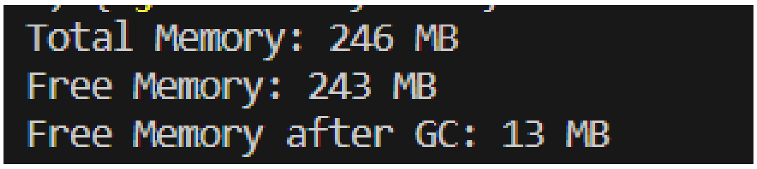
AIM

Write a program to find out total memory, free memory and free memory after executing garbage Collector (gc)..

PROGRAM

```
public class MemoryDemo {
    public static void main(String[] args) {
        // Get the runtime object
        Runtime runtime = Runtime.getRuntime();

        // Display the total memory
        long totalMemory = runtime.totalMemory();
        System.out.println("Total Memory: " +
        totalMemory / (1024 * 1024) + " MB");

        // Display the free memory
        long freeMemory = runtime.freeMemory();
        System.out.println("Free Memory: " +
        freeMemory / (1024 * 1024) + " MB");

        // Run the garbage collector
        runtime.gc();

        // Display free memory after running the garbage collector
        freeMemory = runtime.freeMemory();
        System.out.println("Free Memory after
        GC: " + freeMemory / (1024 * 1024) + " MB");
    }
}
```

SAMPLE INPUT-OUTPUT



```
Total Memory: 246 MB
Free Memory: 243 MB
Free Memory after GC: 13 MB
```

COPY JAVA FILES

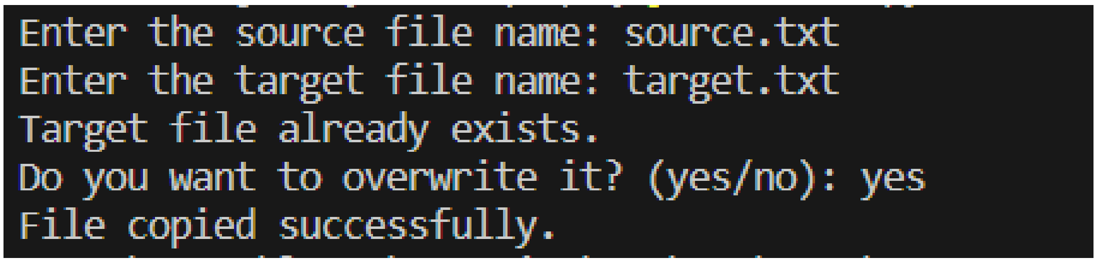
AIM

. Write a program to copy a file to another file using Java to package classes. Get the file names at run time and if the target file exists, then ask for confirmation to overwrite and take necessary actions.

PROGRAM

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.Scanner;

public class FileCopyWithConfirmation {
    public static void main(String[] args) {
        try (Scanner scanner = new Scanner(System.in)) {
            System.out.print("Enter the source file name: ");
            String sourceFileName = scanner.nextLine();

            System.out.print("Enter the target file name: ");
            String targetFileName = scanner.nextLine();

            Path sourcePath = Paths.get(sourceFileName);
            Path targetPath = Paths.get(targetFileName);

            try {
                if (!Files.exists(sourcePath)) {
                    System.out.println("Source file does not exist.");
                } else {
                    if (Files.exists(targetPath)) {
                        System.out.println("Target file
                        already exists.");
                        System.out.print("Do you want to
                        overwrite it? (yes/no): ");
                        String confirmation = scanner.nextLine()
                        .toLowerCase();
                        if (confirmation.equals("yes")) {
                            Files.copy(sourcePath, targetPath,
                            java.nio.file.StandardCopyOption.
                            REPLACE_EXISTING);
                            System.out.println
                            ("File copied successfully.");
                        } else {
                            System.out.println
                            ("File copy canceled.");
                        }
                    } else {
                        Files.copy(sourcePath, targetPath);
                        System.out.println
                        ("File copied successfully.");
                    }
                }
            } catch (IOException e) {
                System.err.println("Error: " + e.getMessage());
            }
        }
    }
}
```

SAMPLE INPUT-OUTPUT



```
Enter the source file name: source.txt
Enter the target file name: target.txt
Target file already exists.
Do you want to overwrite it? (yes/no): yes
File copied successfully.
```

MULTIPTHREADED GUI

AIM

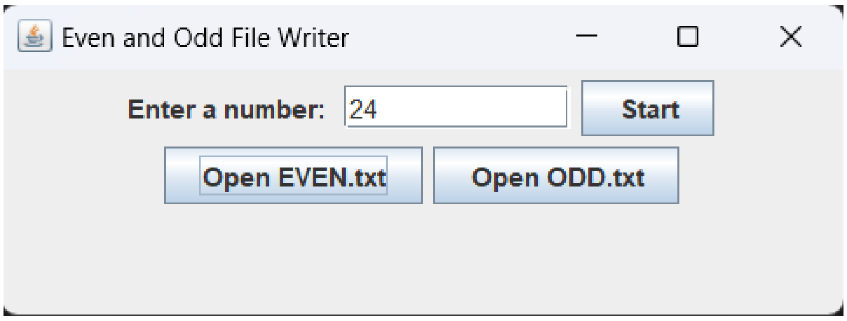
Write a multithreaded GUI java program to write all even numbers less than a given number into a file “EVEN.txt” in one thread, and to write all odd numbers less than a given number into a file “ODD.txt” in another thread. Provide the option to open the created files through GUI

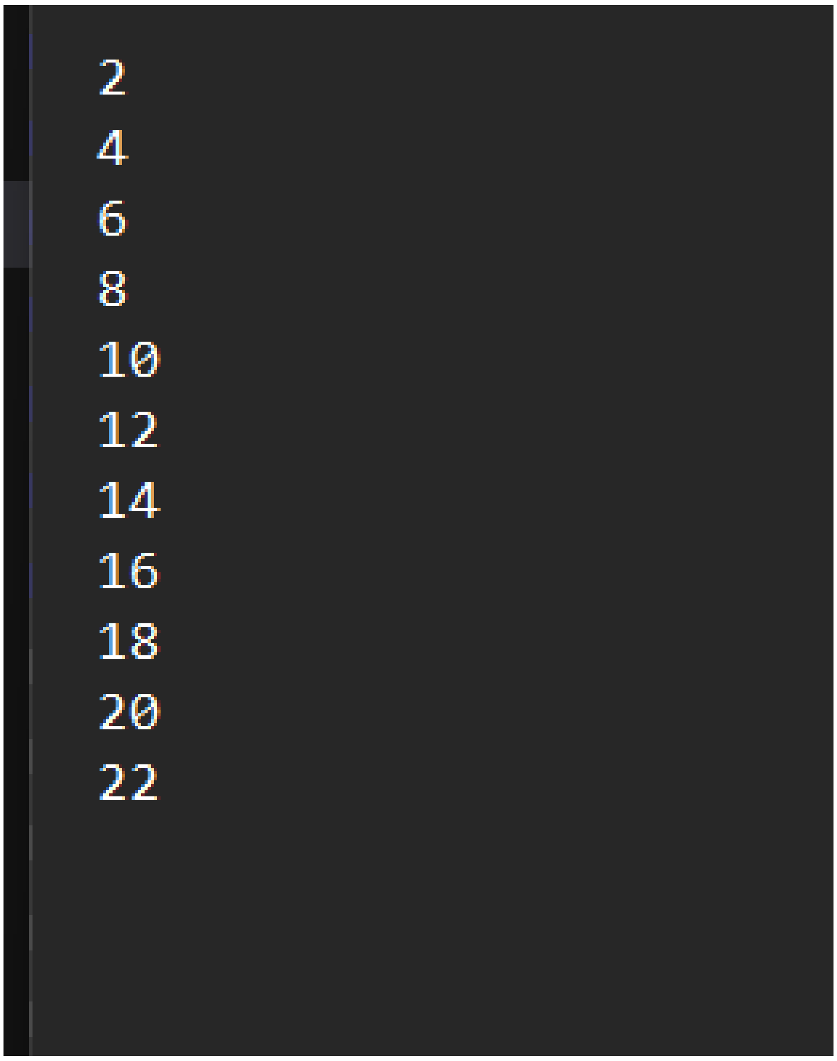
PROGRAM

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

public
class EvenOddFileWriterGUI extends JFrame
{
private
    JTextField inputField;
private
    JButton startButton;
private
    JButton openEvenFileButton;
private
    JButton openOddFileButton;

public
    EvenOddFileWriterGUI()
    {
        setTitle("Even and Odd File Writer");
        setSize(400, 150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Create GUI components
        inputField = new JTextField(10);
        startButton = new JButton("Start");
        openEvenFileButton = new JButton("Open EVEN.txt");
        openOddFileButton = new JButton("Open ODD.txt");
        // Add components to the frame
        JPanel panel = new JPanel();
        panel.add(new JLabel("Enter a number: "));
        panel.add(inputField);
        panel.add(startButton);
        panel.add(openEvenFileButton);
        panel.add(openOddFileButton);

        add(panel);

        // Add action listeners
        startButton.addActionListener(new ActionListener() {
            @Override public void actionPerformed(ActionEvent e)
            {
                int n = Integer.parseInt(inputField.getText());
                Thread evenThread = new Thread(new FileWriteTask
                (n, "EVEN.txt"));
                Thread oddThread = new Thread(new FileWriteTask
                (n, "ODD.txt"));
                evenThread.start();
                oddThread.start();
            }
        });

        openEvenFileButton.addActionListener(new ActionListener() {
            @Override public void actionPerformed(ActionEvent e)
            {
                openFile("EVEN.txt");
            }
        });

        openOddFileButton.addActionListener(new ActionListener() {
            @Override public void actionPerformed(ActionEvent e)
            {
                openFile("ODD.txt");
            }
        });
    }

private
```

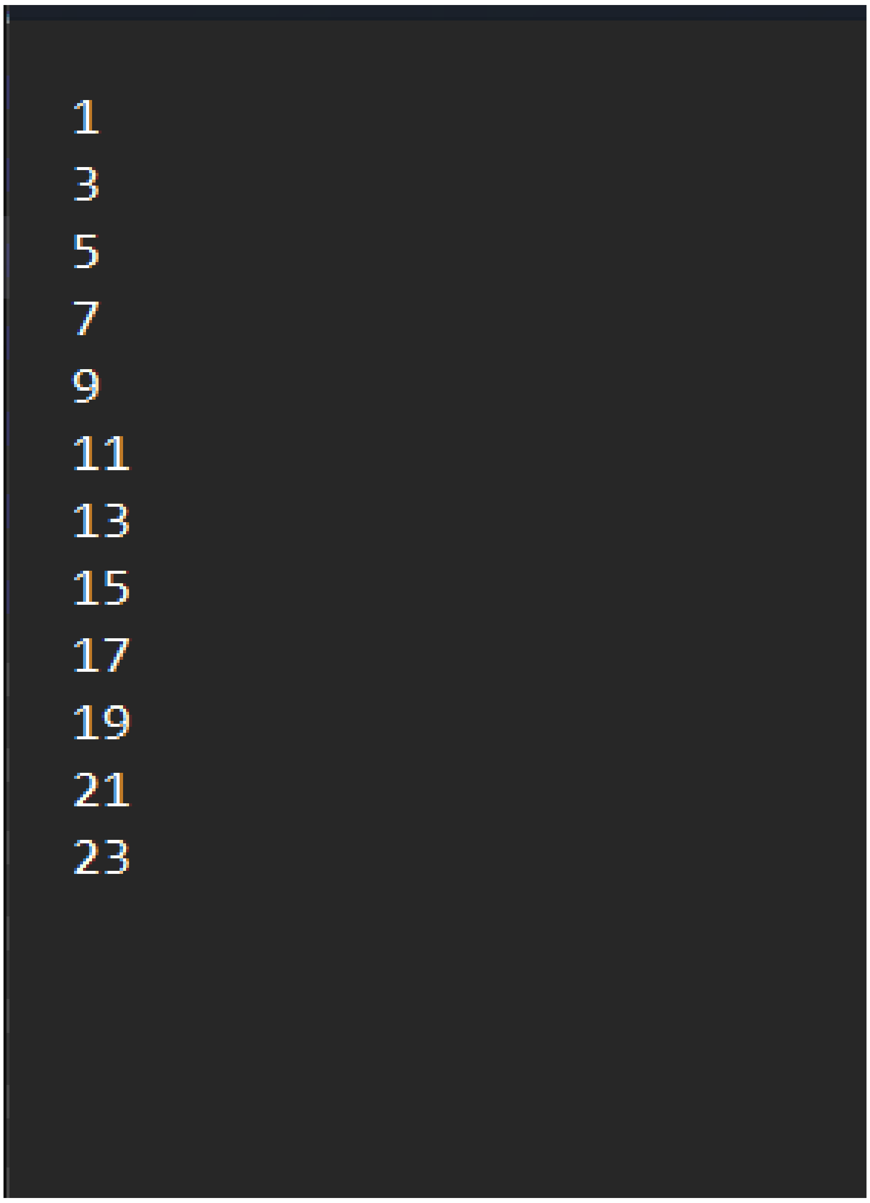
```
    void openFile(String filename)
    {
        try
        {
            Desktop.getDesktop().open(new java.io.File(filename));
        }
        catch (IOException ex)
        {
            JOptionPane.showMessageDialog(this,

            "Error opening the file: " + ex.getMessage());
        }
    }

private
    class FileWriteTask implements Runnable
    {
    private
        int n;
    private
        String filename;

    public
        FileWriteTask(int n, String filename)
        {
            this.n = n;
            this.filename = filename;
        }

        @Override public void run()
        {
            try(FileWriter writer = new FileWriter(filename))
            {
                for (int i = 1; i < n; i++)
                {
                    if (filename.equals("EVEN.txt") && i % 2 == 0)
                    {
                        writer.write(i + "\n");
                    }
                    else if (filename.equals("ODD.txt") && i % 2 != 0)
                    {
                        writer.write(i + "\n");
                    }
                }
            }
            catch (IOException e)
            {
                e.printStackTrace();
            }
        }
    }

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

SAMPLE INPUT-OUTPUT





```
2
4
6
8
10
12
14
16
18
20
22
```



```
1
3
5
7
9
11
13
15
17
19
21
23
```

THREAD SYNCHRONIZATION

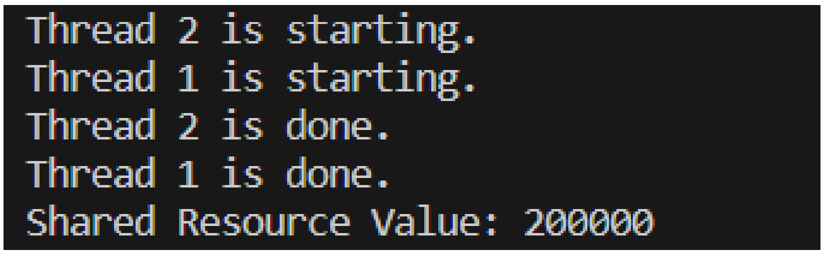
AIM

Write a program to illustrate thread synchronization.

PROGRAM

```
public class ThreadSynchronizationExample {
    public static void main(String[] args) {
        // Create a shared resource
        SharedResource sharedResource = new SharedResource();

        // Create two threads
        Thread thread1 = new Thread(new IncrementTask
        (sharedResource, "Thread 1"));
        Thread thread2 = new Thread(new IncrementTask
        (sharedResource, "Thread 2"));

        // Start the threads
        thread1.start();
        thread2.start();

        try {
            // Wait for both threads to finish
            thread1.join();
            thread2.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        // The final value of the shared resource should be 200,000
        System.out.println("Shared Resource Value: " +
        sharedResource.getValue());
    }
}

class SharedResource {
    private int value = 0;

    // Synchronize the method to ensure only one thread
    can access it at a time
    public synchronized void increment() {
        for (int i = 0; i < 100000; i++) {
            value++;
        }
    }

    public int getValue() {
        return value;
    }
}

class IncrementTask implements Runnable {
    private SharedResource sharedResource;
    private String threadName;

    public IncrementTask(SharedResource sharedResource, String threadName) {
        this.sharedResource = sharedResource;
        this.threadName = threadName;
    }

    @Override
    public void run() {
        System.out.println(threadName + " is starting.");
        sharedResource.increment();
        System.out.println(threadName + " is done.");
    }
}
```

SAMPLE INPUT-OUTPUT



```
Thread 2 is starting.
Thread 1 is starting.
Thread 2 is done.
Thread 1 is done.
Shared Resource Value: 200000
```

ANALOG CLOCK

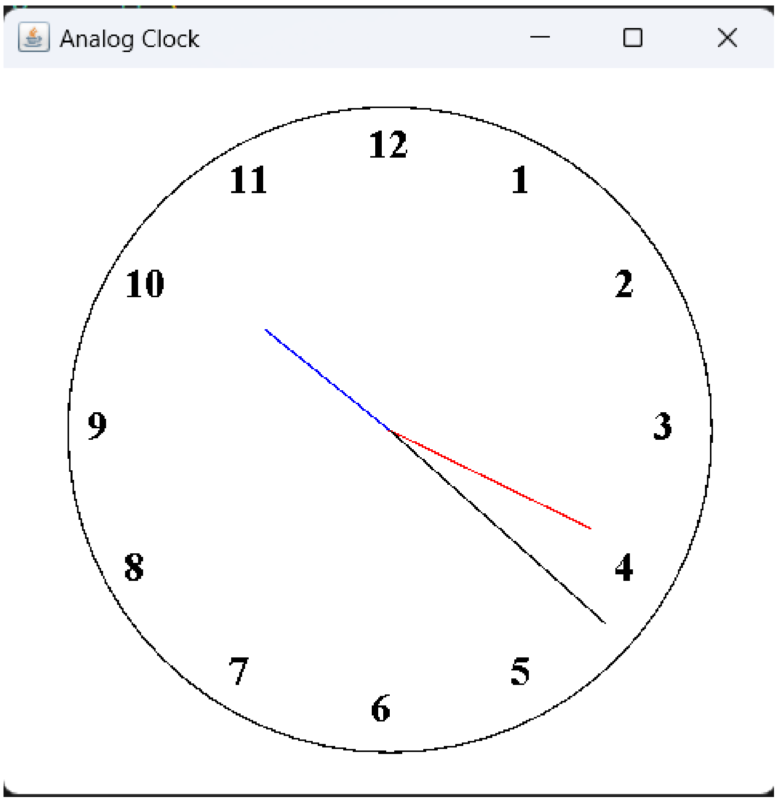
AIM

Write an applet program to create an analog clock. Receive the starting time as an applet parameter

PROGRAM

```
import javax.swing.*;
import java.awt.*;
import java.text.SimpleDateFormat;
import java.util.Calendar;

public class AnalogClockApp {
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> createAndShowGUI());
    }

    private static void createAndShowGUI() {
        JFrame frame = new JFrame("Analog Clock");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 400);

        AnalogClockPanel clockPanel = new AnalogClockPanel();
        frame.add(clockPanel);

        Timer timer = new Timer(1000, e -> clockPanel.repaint());
        timer.start();

        frame.setVisible(true);
    }
}

class AnalogClockPanel extends JPanel {
    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);

        int centerX = getWidth() / 2;
        int centerY = getHeight() / 2;
        int radius = Math.min(centerX, centerY) - 20;
        g.setColor(Color.WHITE);
        g.fillRect(0, 0, getWidth(), getHeight());

        // Draw clock face
        g.setColor(Color.BLACK);
        g.drawOval(centerX - radius, centerY - radius,

        radius * 2, radius * 2);

        // Draw clock numbers
        g.setFont(new Font("Serif", Font.BOLD, 20));
        for (int i = 1; i <= 12; i++) {
            double angle = Math.toRadians(90 - i * 30);
            int numX = (int) (centerX + (radius - 20) * Math.cos(angle));
            int numY = (int) (centerY - (radius - 20) * Math.sin(angle));
            g.drawString(Integer.toString(i), numX - 10, numY + 5);
        }

        // Get current time
        Calendar calendar = Calendar.getInstance();
        int hours = calendar.get(Calendar.HOUR_OF_DAY);
        int minutes = calendar.get(Calendar.MINUTE);
        int seconds = calendar.get(Calendar.SECOND);

        // Draw clock hands
        drawClockHands(g, centerX, centerY, radius, hours, minutes, seconds);
    }

    private void drawClockHands(Graphics g, int centerX,
    int centerY, int radius, int hours, int minutes, int seconds) {
        double hourAngle = Math.toRadians(90 - (hours % 12) * 30 - minutes / 2);
        double minuteAngle = Math.toRadians(90 - minutes
        * 6 - seconds / 10);
        double secondAngle = Math.toRadians(90 - seconds
        * 6);

        int hourHandLength = (int) (radius * 0.5);
        int minuteHandLength = (int) (radius * 0.7);
        int secondHandLength = (int) (radius * 0.9);
        int hourHandX = centerX + (int) (hourHandLength
        * Math.cos(hourAngle));
        int hourHandY = centerY - (int) (hourHandLength
        * Math.sin(hourAngle));
        int minuteHandX = centerX + (int) (minuteHandLength
        * Math.cos(minuteAngle));
        int minuteHandY = centerY - (int) (minuteHandLength
        * Math.sin(minuteAngle));
        int secondHandX = centerX + (int) (secondHandLength
        * Math.cos(secondAngle));
        int secondHandY = centerY - (int) (secondHandLength
        * Math.sin(secondAngle));

        g.setColor(Color.BLUE);
        g.drawLine(centerX, centerY, hourHandX, hourHandY);
        g.setColor(Color.RED);
        g.drawLine(centerX, centerY, minuteHandX, minuteHandY);
        g.setColor(Color.BLACK);
        g.drawLine(centerX, centerY, secondHandX, secondHandY);
    }
}
```

SAMPLE INPUT-OUTPUT



TOURISM MANAGEMENT SYSTEM

PROJECT REPORT

INTRODUCTION

Data is a collection of facts and figures that represent recordable information. Databases are organized collections of related data, and they help produce valuable information. For instance, data on students' marks can be used to determine toppers and averages. A Database Management System (DBMS) stores data efficiently, making it easy to retrieve and manipulate.

SQL, or Structured Query Language, is a specific programming language for managing data in relational databases. It's used widely for data storage, retrieval, and management.

JDBC, or Java-Database Connectivity, is a Java technology that links Java applications with relational databases. It facilitates data operations in Java applications by acting as a bridge between them and databases.

The Tourism Management System is a software tool designed to streamline hostel administration. It automates tasks like student registration, room allocation, billing, and maintenance tracking. Database connectivity is vital in this system, allowing efficient data storage and retrieval.

This report explores the Tourism Management System's significance, objectives, and features, with a focus on its use of database connectivity. It also includes Java code snippets to illustrate this connection.

USERS AND USER REQUIREMENTS

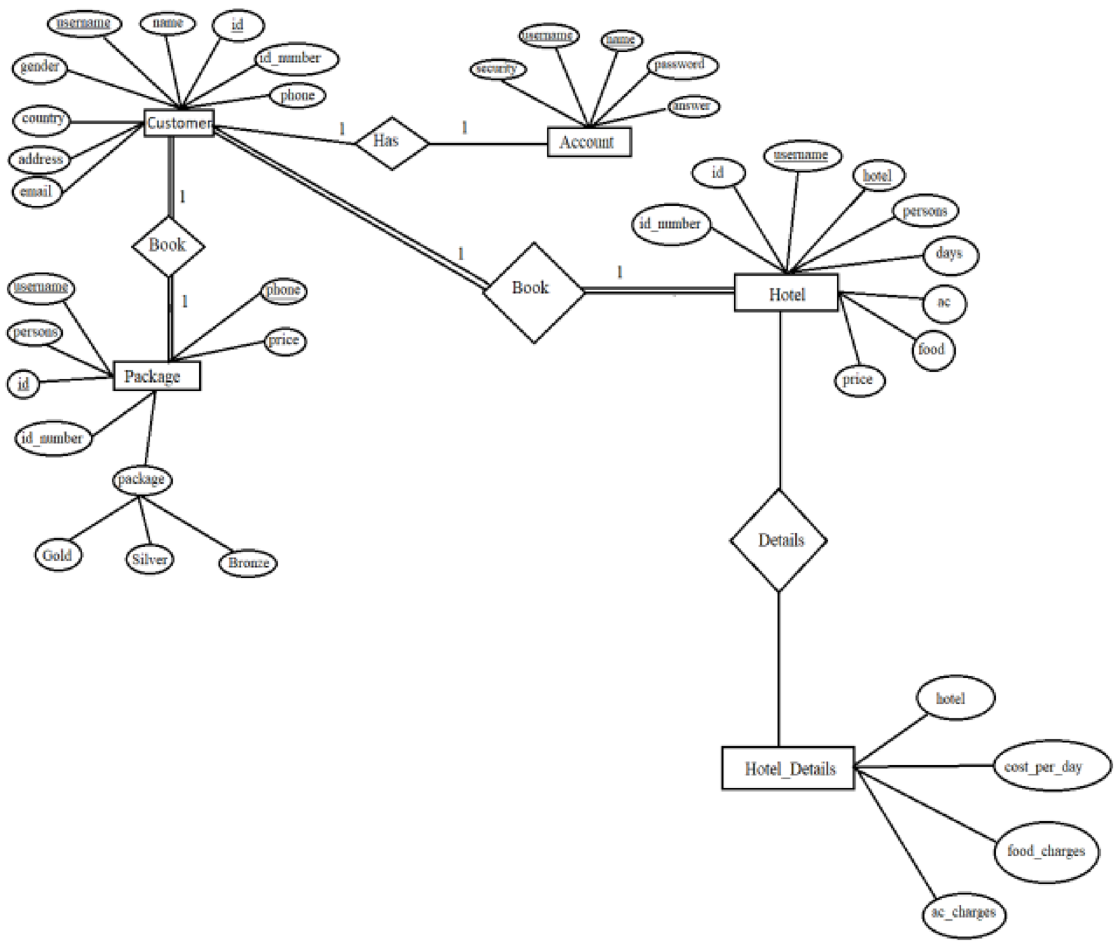
Users in need of a tourism booking system encompass a diverse group of individuals, each with specific requirements and expectations. Among them are tourists and travelers, ranging from solo explorers to families and luxury seekers, all seeking secure and convenient accommodation options. Students, whether for educational purposes, internships, or exchange programs, rely on such systems for affordable lodging. Business travelers, in their pursuit of cost-effective stays with essential amenities, also fall into this category. Event attendees, budget-conscious travelers, and backpackers looking to maximize their journey within budget constraints all depend on hostel booking systems. On the service side, hostel owners and managers utilize these systems to efficiently manage reservations and guest information. Travel agencies and tourism authorities, aiming to offer diverse accommodations or promote local hostels, use these systems. Additionally, developers and tech enthusiasts may integrate hotel booking systems into their platforms, emphasizing the need for compatibility and user-friendly features.

Understanding this wide array of users is pivotal in designing a comprehensive and accessible hotel booking system that caters to the varied needs of travelers and accommodation providers alike.

In the contemporary world, tourism is a vital and lucrative industry, offering cultural exposure and economic benefits. Travelers often seek secure accommodations in unfamiliar destinations. Our program simplifies this by enabling advance lodging reservations, ensuring a smooth and enjoyable trip.

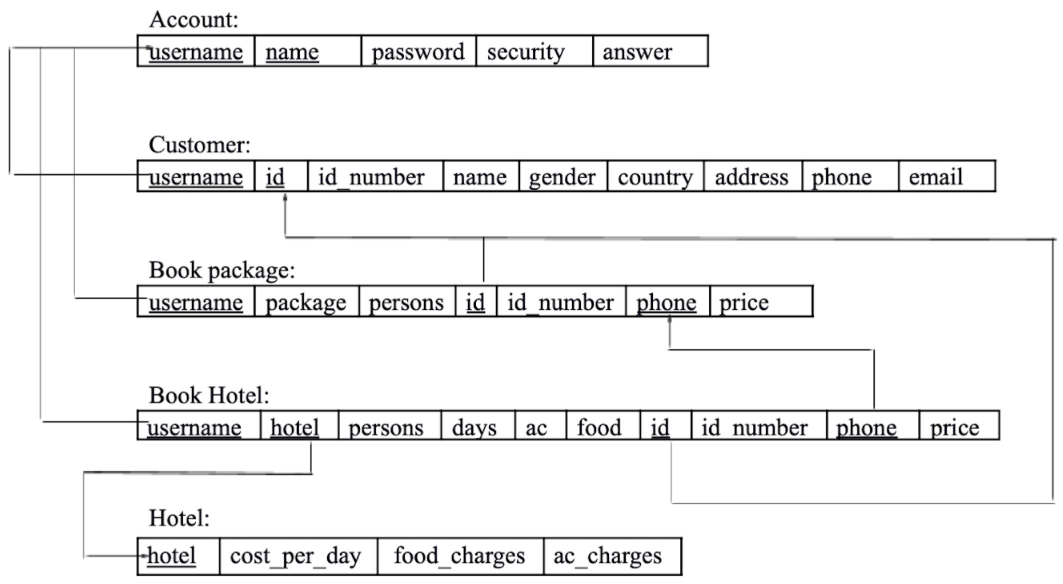
This report explores our program's features and benefits, emphasizing its role in enhancing the travel experience and boosting the tourism sector's economic vitality. We are committed to making travel more accessible and memorable for all.

UML DIAGRAM



The main objective of the Tourism Management System is to manage the details of Customer, Hotel Booking, Cancellation and Tourism places. It manages all the information about Users, Hotel, Packages etc. The project is totally built at administrative end and thus only the administrator is guaranteed the access to the backend database. The purpose of this project is to build an application program to reduce the manual work for managing Tourists, Booking, Places etc.
This application will help in accessing the information related to the travel to the particular destination with great ease. The users can track the information related to their tours with great ease through this application. The travel agency information can also be obtained through this application.
Through this system, the propose system is highly automated and makes the travelling activities much easier and flexible. The user can get the very right information at the very right time. This system will include all the necessary fields which are required during online reservation time. This system will be easy to use and can be used by any person. The basic idea behind this project is to save data in a central database which can be accessed by any authorize person to get information and saves time and burden which are being faced by their customers.
Administrator can access and modify the information stored in the database of this system, this includes adding and updating of details, and it will give accurate information and simplifies manual work and also it minimizes the documentation related work. Provides up to date information. Finally booking confirmation notification will be send to the users.

Tourists can register by providing personal details, make new reservation and book only one hotel and package and can make cancellation.



DBMS FUNCTIONING(TABLE STRUCTURES)

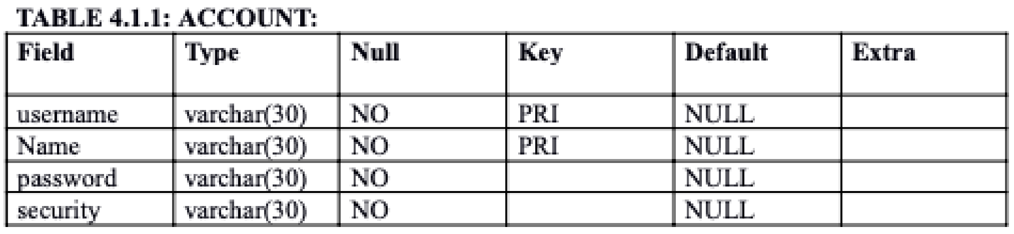


TABLE 4.1.1: ACCOUNT:

Field	Type	Null	Key	Default	Extra
username	varchar(30)	NO	PRI	NULL	
Name	varchar(30)	NO	PRI	NULL	
password	varchar(30)	NO		NULL	
security	varchar(30)	NO		NULL	

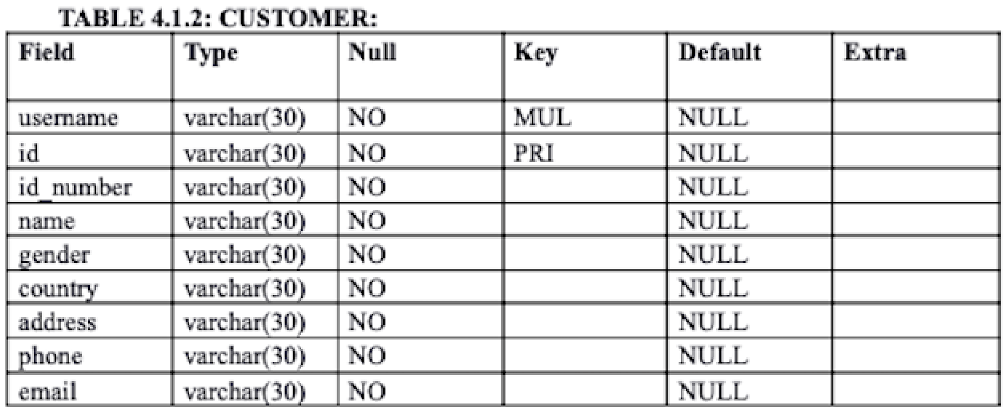


TABLE 4.1.2: CUSTOMER:

Field	Type	Null	Key	Default	Extra
username	varchar(30)	NO	MUL	NULL	
id	varchar(30)	NO	PRI	NULL	
id_number	varchar(30)	NO		NULL	
name	varchar(30)	NO		NULL	
gender	varchar(30)	NO		NULL	
country	varchar(30)	NO		NULL	
address	varchar(30)	NO		NULL	
phone	varchar(30)	NO		NULL	
email	varchar(30)	NO		NULL	

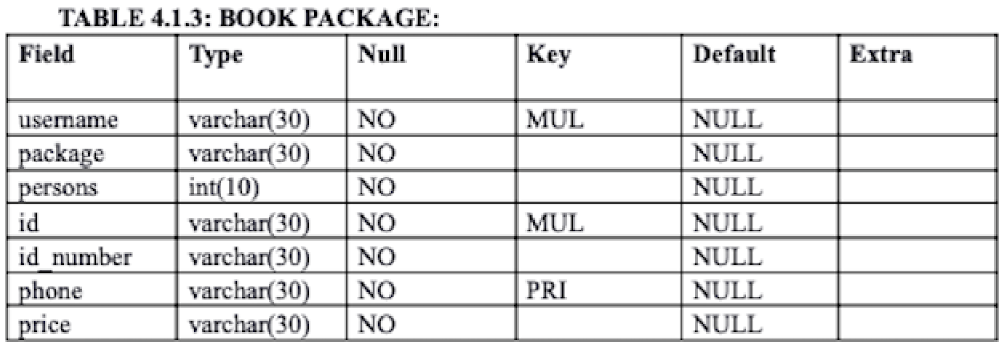


TABLE 4.1.3: BOOK PACKAGE:

Field	Type	Null	Key	Default	Extra
username	varchar(30)	NO	MUL	NULL	
package	varchar(30)	NO		NULL	
persons	int(10)	NO		NULL	
id	varchar(30)	NO	MUL	NULL	
id_number	varchar(30)	NO		NULL	
phone	varchar(30)	NO	PRI	NULL	
price	varchar(30)	NO		NULL	

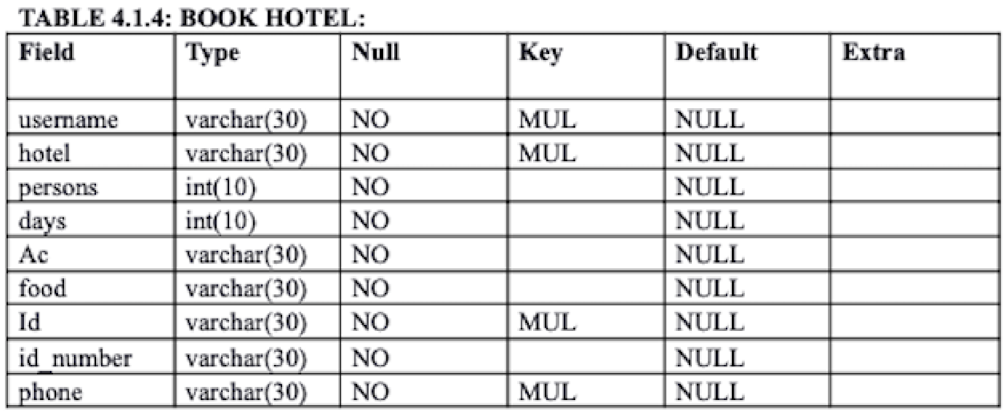
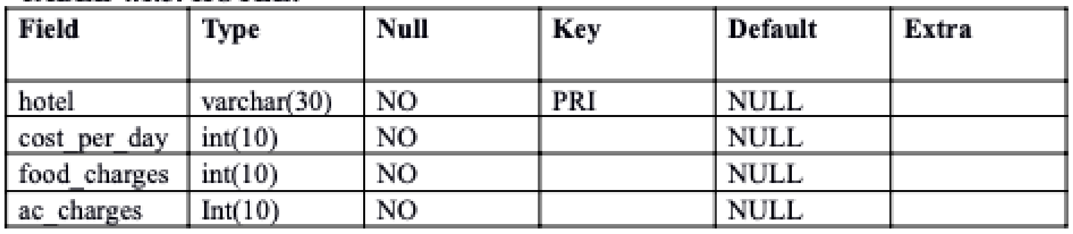


TABLE 4.1.4: BOOK HOTEL:

Field	Type	Null	Key	Default	Extra
username	varchar(30)	NO	MUL	NULL	
hotel	varchar(30)	NO	MUL	NULL	
persons	int(10)	NO		NULL	
days	int(10)	NO		NULL	
Ac	varchar(30)	NO		NULL	
food	varchar(30)	NO		NULL	
Id	varchar(30)	NO	MUL	NULL	
id_number	varchar(30)	NO		NULL	
phone	varchar(30)	NO	MUL	NULL	

TABLE 4.1.5: HOTEL:



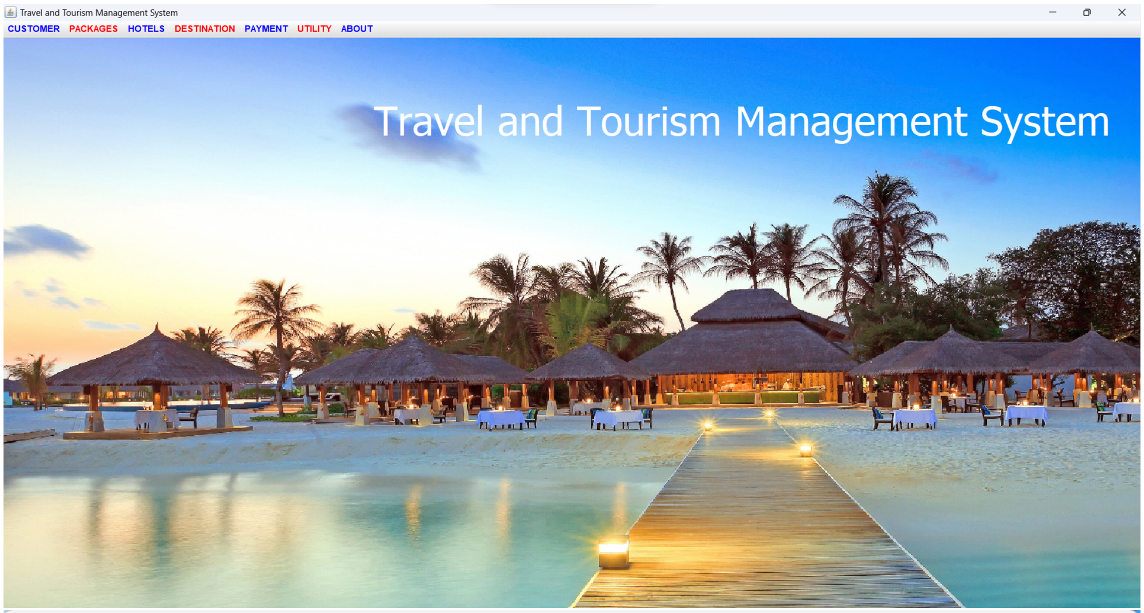
Field	Type	Null	Key	Default	Extra
hotel	varchar(30)	NO	PRI	NULL	
cost_per_day	int(10)	NO		NULL	
food_charges	int(10)	NO		NULL	
ac_charges	Int(10)	NO		NULL	

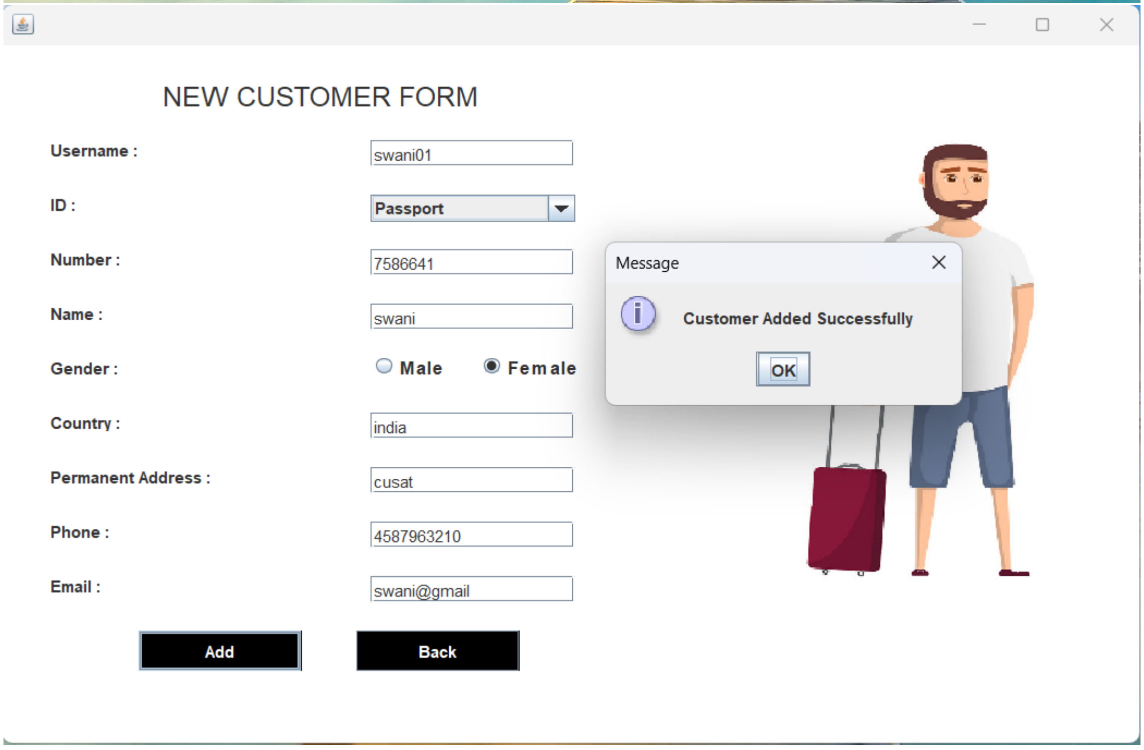
IMPLEMENTATION USING JAVA FEATURES

- Object-Oriented Programming (OOP) - Java is known for its strong support for OOP principles. Classes and objects can be used to model the entities in the system, such as customers, bookings, and destinations.
- Inheritance - Inheritance allows the creation of hierarchical relationships between classes. It can be used to model and implement common properties and behaviors among different entities in the system, making the code more organized and maintainable.
- Polymorphism - Polymorphism enables the use of different classes through a common interface or base class. It's beneficial for handling various types of data, like different types of bookings or tourist destinations.
- Encapsulation - Encapsulation involves the bundling of data and methods into classes. This feature helps in maintaining data integrity and controlling access to critical components, such as user profiles or booking information.
- Long - Abstraction - Abstraction allows developers to hide the complex implementation details and provide a simplified interface. In the context of your project, it could be used to create high-level functions or methods for managing customer data, bookings, and tourism places.
- Exception Handling - Java provides a robust mechanism for handling exceptions, ensuring that the system can gracefully deal with unexpected errors or issues during execution. For example, when there's an issue with booking a hotel, the system can handle it and provide appropriate feedback to users.
- Collections Framework - Java's collections framework offers a wide range of data structures like lists, sets, and maps. These can be used to manage and manipulate data efficiently, such as maintaining lists of customers, bookings, and places.
- Java Database Connectivity (JDBC) - Since your project involves managing data, JDBC is a vital Java feature. It allows your application to interact with a relational database, facilitating operations like storing customer information, handling reservations, and managing tourism places data.

- User Interface (UI) Development - Java provides libraries and frameworks for building user interfaces. JavaFX or Swing, for instance, can be used to create a user-friendly interface for administrators and travelers to interact with the system.
- Multithreading - In a project where multiple users might access and manipulate data concurrently, multithreading can be useful. It enables the system to handle multiple requests simultaneously, ensuring responsiveness and efficient resource utilization.

RESULTS-SCREENSHOTS


Travel and Tourism Management System
CUSTOMER
PACKAGES
HOTELS
DESTINATION
PAYMENT
UTILITY
ABOUT
Travel and Tourism Management System

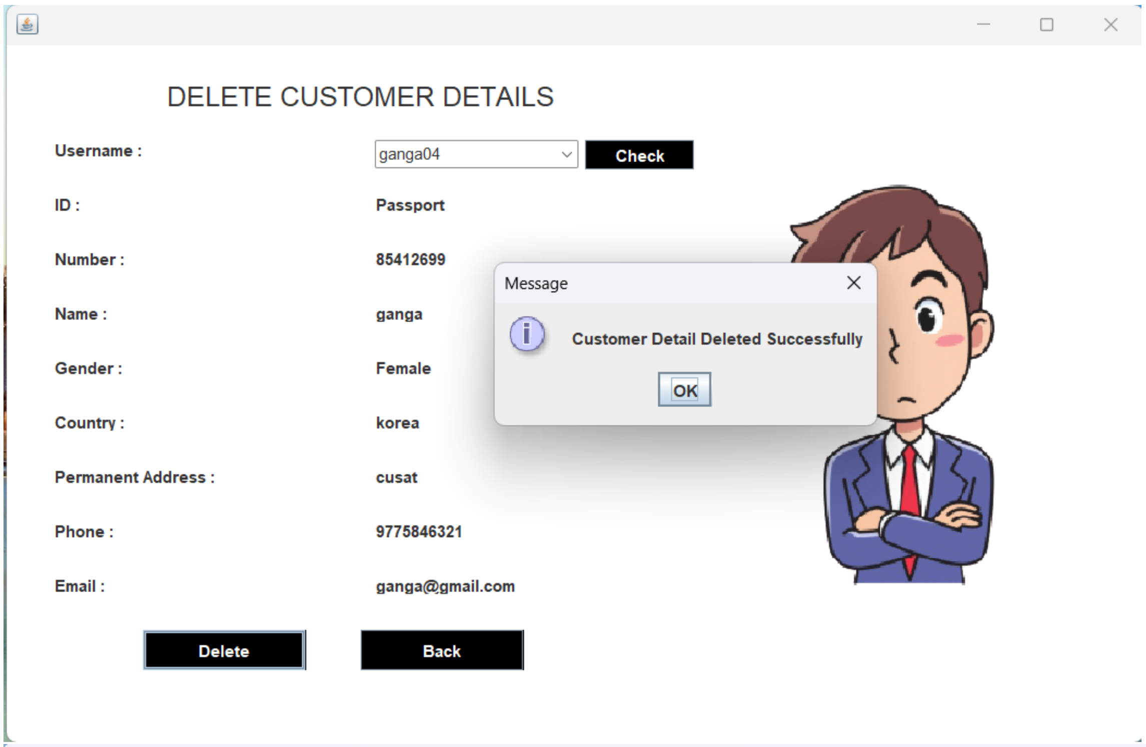


NEW CUSTOMER FORM

Username :	swani01
ID :	Passport
Number :	7586641
Name :	swani
Gender :	Male [✓ Female]
Country :	india
Permanent Address :	cusat
Phone :	4587963210
Email :	swani@gmail

Message

Customer Added Successfully

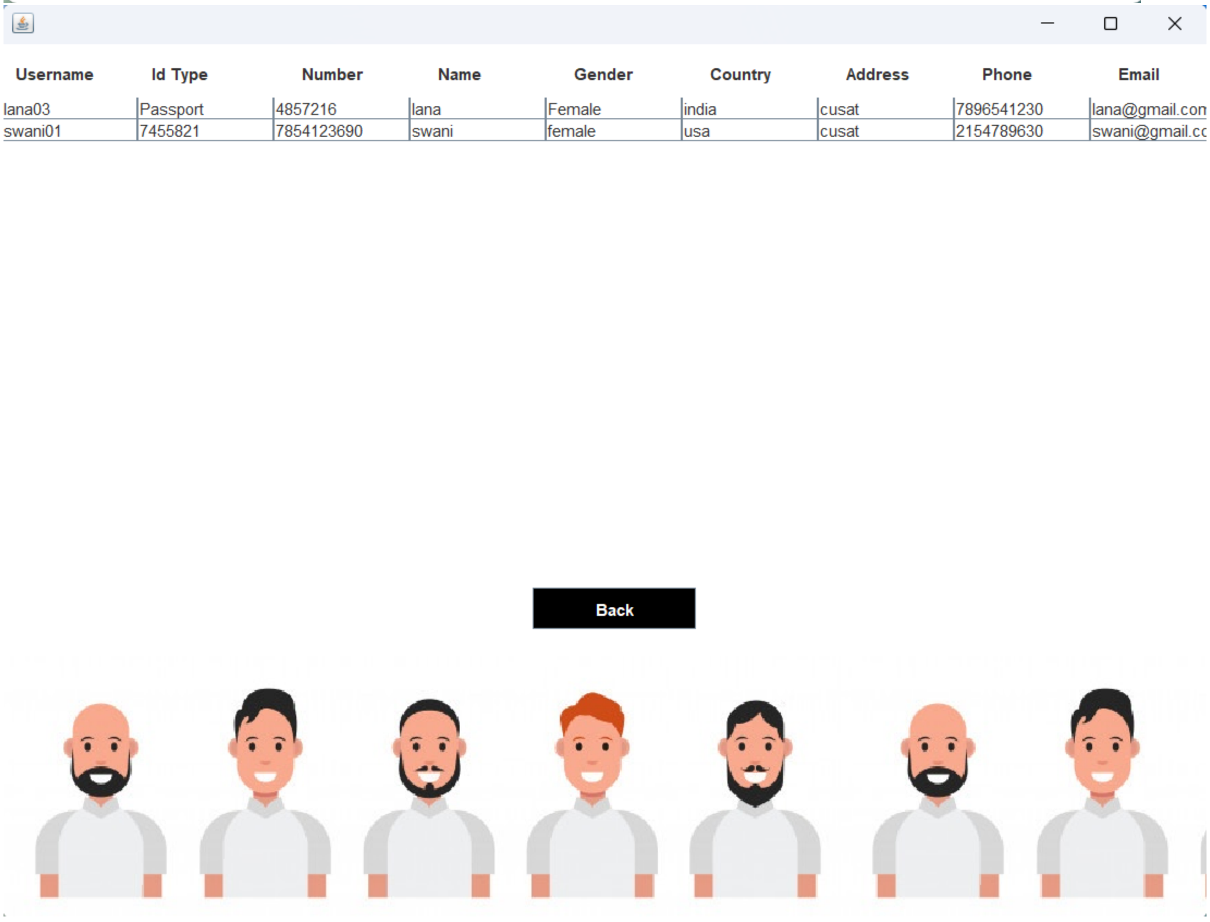


DELETE CUSTOMER DETAILS

Username :	ganga04
ID :	Passport
Number :	85412699
Name :	ganga
Gender :	Female
Country :	korea
Permanent Address :	cusat
Phone :	9775846321
Email :	ganga@gmail.com

Message

Customer Detail Deleted Successfully



Username	Id Type	Number	Name	Gender	Country	Address	Phone	Email
lana03	Passport	4857216	lana	Female	india	cusat	7896541230	lana@gmail.com
swani01	7455821	7854123690	swani	female	usa	cusat	2154789630	swani@gmail.cc

CONCLUSION

In conclusion, the Tourism Management System represents an innovative solution aimed at streamlining the intricate world of travel and accommodation management. This project not only simplifies the complex task of overseeing customers, bookings, cancellations, and tourism destinations but also enhances the user's travel experience by providing a user-friendly and efficient platform. The system caters to both administrators, who gain comprehensive control and access to a centralized database, and travelers, who can effortlessly make reservations and track their journeys. By reducing manual efforts, improving access to accurate information, and facilitating seamless communication, this project is poised to significantly impact the tourism industry, ensuring that each journey is a smooth and enjoyable experience for all involved