DEPARTMENT OF COMPUTER SCIENCE
RAJAGIRI COLLEGE OF SOCIAL SCIENCES
(Autonomous)
KALAMASSERY - KOCHI - 683104

MASTER OF COMPUTER APPLICATIONS

JAVA PROGRAMMING LAB RECORD

NAME : **SARATH CHANDRAN M**

SEMESTER : **SECOND SEMESTER**

REGISTER NO. : **23204051**

DEPARTMENT OF COMPUTER SCIENCE
RAJAGIRI COLLEGE OF SOCIAL SCIENCES
(Autonomous)
KALAMASSERY - KOCHI – 683104

MASTER OF COMPUTER APPLICATIONS

CERTIFICATE

NAME : SARATH CHANDRAN M

SEMESTER : SECOND

REGISTER NO. : 23204051

*Certified that this is a bonafide record of work done by the **Sarath Chandran M** in the Software Laboratory of Department of Computer Science, Rajagiri College of Social Sciences, Kalamassery.*

Ms. Neethu Narayanan
Faculty in Charge

Dr. Bindiya M Varghese
Dean, Computer Science

Internal Examiner

External Examiner

Place : Kalamassery
Date :

Table of Contents

Activity **Page No**

CYCLE 01:	**Date: 18-01-2024**
Basic programs using datatypes, operators and control statement in java	

1. Write a Java program to check whether a string is palindrome or not.

CODE:

```
import java.io.*;
class PalindromeChecker{
   public static void main(String args[]){
      DataInputStream din=new DataInputStream(System.in);
      String str, revStr="";
      try{
        System.out.println("Enter a string:");
        str=din.readLine();
        int strLen = str.length();
        for(int i=(strLen-1);i>=0;i--
        ){
           revStr = revStr + str.charAt(i);
        }
        str = str.toLowerCase();
        revStr =
        revStr.toLowerCase();
        if(str.equals(revStr)){
           System.out.println(" The string is a palindrome");
        }
        else{
           System.out.println(" The string is not a palindrome");
        }
      }
      catch(Exception e){
        System.out.println(e);
      }
   }
}
```

OUTPUT:

```
C:\Users\admin\Desktop\java>java PalindromeChecker
Enter a string: mom
The string is a palindrome.
```

2. Write a Java program to multiply two matrices.

CODE:

```
import java.io.*;
import java.util.*;
class prgm2{
   public static void main(String[] args)
      {int a[][],b[][],res[][];
      int r,c,i,j,k;
      Scanner s = new
      Scanner(System.in);try{
         System.out.println("Enter the number of rows and columns : ");
         r=s.nextInt();
         c=s.nextInt();
         a=new int[r][c];
         b=new int[r][c];
         res=new int[r][c];
         System.out.println("Enter the first matrix : ");
         for(i=0;i<r;++i){
            for(j=0;j<c;j++){
               a[i][j]=s.nextInt();
            }
         }
         System.out.println("Enter the second matrix : ");
         for(i=0;i<r;++i){
            for(j=0;j<c;j++){
               b[i][j]=s.nextInt();
            }
         }
         for(i=0;i<r;++i){
            for(j=0;j<c;j++
            ){
               res[i][j]=0;
               for(k=0;k<r;k++){
                  res[i][j] = res[i][j] + a[i][k] * b[k][j];          }
            }
        }
         System.out.println("First Matrix :
         ");for(i=0;i<r;++i){
            for(j=0;j<c;j++){
               System.out.print(a[i][j]+"
               ");
            }
            System.out.println("");
         }
         System.out.println("Second Matrix : ");
         for(i=0;i<r;++i){
            for(j=0;j<c;j++){
               System.out.print(b[i][j]+"");
```

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

        System.out.println("Result : ");
        for(i=0;i<r;++i){
            for(j=0;j<c;j++){
                System.out.print(res[i][j]+"  ");
            }
            System.out.println("");
        }
      }catch(Exception e){
        System.out.println("Error : "+e);
      }

   }
}
```

OUTPUT:

```
C:\Users\cclab25\Desktop\javaa>java prgm2
Enter the number of rows and columns :
2
2
Enter the first matrix :
3 7
9 2
Enter the second matrix :
2 6
3 4
First Matrix :
3  7
9  2
Second Matrix :
2  6
3  4
Result :
27  46
24  62
```

3. Write a Java program to find the transpose of a matrix.

CODE:

```
import java.io.*;
public class prgm3 {
   public static void main(String args[]){
      DataInputStream din= new DataInputStream(System.in);
      int a[][],r,c,t[][];
      try{
         System.out.println("Enter the no of rows:");
         r=Integer.parseInt(din.readLine());
         System.out.println("Enter the no of columns:");
         c=Integer.parseInt(din.readLine());
         a= new int[r][c];
         t= new int[r][c];
         for(int
         i=0;i<r;i++){
            for(int  j=0;j<c;j++){
               System.out.println("Enter a["+i+"]["+j+"]");
               a[i][j]=Integer.parseInt(din.readLine());
            }
         }
         for(int i=0;i<r;i++){
            for(int
            j=0;j<c;j++){
               t[i][j]=a[j][i];
            }
         }
         System.out.println("Matrix:\n");
         for(int i=0;i<r;i++){
            for(int j=0;j<c;j++){
               System.out.print(a[i][j]+"
               ");
            }
            System.out.println("");
         }
         System.out.println("Transpose:\n");
         for(int i=0;i<r;i++){
            for(int j=0;j<c;j++){
               System.out.print(t[i][j]+"
               ");
            }
            System.out.println("");
         }
      }
      catch(Exception e){
         System.out.println(e);
      }
   } }
```

Output:

```
C:\Users\cclab25\Desktop\javaa>java prgm3
Enter the no of rows:
2
Enter the no of columns:
2
Enter a[0][0]
3
Enter a[0][1]
2
Enter a[1][0]
8
Enter a[1][1]
6
Matrix:

3 2
8 6
Transpose:

3 8
2 6
```

4 Write a Java program to find the second smallest element in an array.

CODE:

```
import java.io.*;
import java.util.*;
public class prgm4 {
   public static void main(String[] args)
      {int a[];
      int i,j,n,temp;
      Scanner s = new
      Scanner(System.in);try{
         System.out.println("Enter n :");
         n=s.nextInt();
         a=new int[n];
         System.out.println("Enter the elements :");
         for(i=0;i<n;i++){
            a[i]=s.nextInt();
         }

         for(i=0;i<n;i++){
            for(j=0;j<n;j++
            )
            {
               if(a[i] <
                  a[j]){
                  temp=a[i];
                  a[i]=a[j];
                  a[j]=temp;
               }
            }
         }

         System.out.println("Second  Smallest Element is "+a[1]);
      }catch(Exception e){
         System.out.println("Error : "+e);
      }
   }
}
```

OUTPUT:

```
C:\Users\cclab25\Desktop\javaa>javac prgm4.java

C:\Users\cclab25\Desktop\javaa>java prgm4
Enter n :
5
Enter the elements :
3
7
9
2
5
Second  Smallest Element is 3
```

5. Write a Java program to check whether a number is prime or not.

CODE:

```
import java.util.*;
import java.io.*;
class prgm5_c1{
   public static void main(String args[]){
      System.out.println("Enter the number:");
      Scanner sc=new Scanner(System.in);
      int num=sc.nextInt();
      boolean flag=false;
      for(int
      i=2;i<=num/2;++i){
         if(num%i==0)
            {flag=true;
            break;
         }
      }
      if(!flag){
         System.out.println(num +" is a prime number");
      }
      else{
         System.out.println(num +" is not a prime number");
      }
   }
}
```

OUTPUT:

```
C:\Users\cclab25\Desktop\javaa>javac prgm5.java

C:\Users\cclab25\Desktop\javaa>java prgm5
Enter the number:
19
19 is a prime number

C:\Users\cclab25\Desktop\javaa>java prgm5
Enter the number:
8
8 is not a prime number
```

6 .Write a java program to demonstrate Bitwise logical operators, left shift and right shiftoperators.

CODE:

```
import java.util.*;
import java.io.*;
public class prgm6_c1 {
   public static void main(String[] args)
      {int a;
      int b;
      Scanner sc=new
      Scanner(System.in);
      System.out.println("enter a:");
      a=sc.nextInt();
      System.out.println("enter b:");
      b=sc.nextInt();

      int bitwiseAnd = a & b;
      System.out.println("Bitwise AND: " +
      bitwiseAnd);

      int bitwiseOr = a | b;
      System.out.println("Bitwise OR: " +
      bitwiseOr);

      int bitwiseXor = a ^ b;
      System.out.println("Bitwise XOR: " +
      bitwiseXor);

      int leftShift = a << 1;
      System.out.println("Left Shift: " + leftShift);

      int rightShift = b >> 1;
      System.out.println("Right Shift: " + rightShift);
   }
}
```

OUTPUT:

```
C:\Users\cclab25\Desktop\javaa>javac prgm6_c1.java

C:\Users\cclab25\Desktop\javaa>java prgm6_c1
enter a:
8
enter b:
5
Bitwise AND: 0
Bitwise OR: 13
Bitwise XOR: 13
Left Shift: 16
Right Shift: 2
```

7 .Write a java program to find the roots of a quadratic equation.

CODE:

```
import java.util.*;
public class RootsQuadratic {

   public static void main(String[] args) {
      Scanner s = new
      Scanner(System.in);try {
         double a , b , c;
         double root1, root2;
         System.out.println("Enter the value of a, b and c : ");a
          = s.nextDouble();
         b =
         s.nextDouble();c =
         s.nextDouble();
         double determinant = b * b - 4 * a * c;
         if (determinant > 0) {

          root1 = (-b + Math.sqrt(determinant)) / (2 * a);
          root2 = (-b - Math.sqrt(determinant)) / (2 * a);
          System.out.format("root1 = %.2f and root2 = %.2f", root1, root2);
         }

         else if (determinant == 0) {

          root1 = root2 = -b / (2 * a);
          System.out.format("root1 = root2 = %.2f;", root1);
         }

         else {

          double real = -b / (2 * a);
          double imaginary = Math.sqrt(-determinant) / (2 * a);
          System.out.format("root1 = %.2f+%.2fi", real, imaginary);
          System.out.format("\nroot2 = %.2f-%.2fi", real, imaginary);
         }
      }
      catch (Exception e) {
         System.out.println(e);
      }
   }

   }
```

OUTPUT:

```
C:\Users\cclab25\Desktop\javaa>javac prgm7_c1.java

C:\Users\cclab25\Desktop\javaa>java prgm7_c1
Enter the value of a, b and c :
7
9
4
root1 = -0.64+0.40i
root2 = -0.64-0.40i
```

CYCLE 2:	**Date: 23-01-2024**
Object Oriented Concepts	

1. Write a Java program to calculate the area of different shapes namely circle, rectangle,trapezoid and triangle. (Use the concepts of JAVA like this keyword, constructor overloading and method overloading)

CODE:

```
public class AreaCalculator {
   public static void main(String[] args)
      {Circle circle = new Circle(5);
      Rectangle rectangle = new Rectangle(4, 6);
      Trapezoid trapezoid = new Trapezoid(5, 7,
      9);Triangle triangle = new Triangle(3, 4, 5);
      System.out.println("Area of Circle: " + circle.calculateArea());
      System.out.println("Area of Rectangle: " + rectangle.calculateArea());
      System.out.println("Area of Trapezoid: " + trapezoid.calculateArea());
      System.out.println("Area of Triangle: " + triangle.calculateArea());
   }
}
class Circle {
   private double radius;
   public Circle(double radius)
      {this.radius = radius;
   }
   public double calculateArea() {
      return Math.PI * radius * radius;
   }
}
class Rectangle {
   private double length;
   private double width;
   public Rectangle(double length, double width) {
      this.length = length;
      this.width = width;
   }
   public double calculateArea()
      {return length * width;
   }
}
class Trapezoid  {
   private double base1;
   private double base2;
```

```
    private double height;
    public Trapezoid(double base1, double base2, double height) {
        this.base1 = base1;
        this.base2 = base2;
        this.height = height;
    }
    public double  calculateArea()  {
        return (base1 + base2) * height / 2;
    }
}
class Triangle {
    private double side1;
    private double side2;
    private double side3;
    public Triangle(double side1, double side2, double side3)
        {this.side1 = side1;
        this.side2 = side2;
        this.side3 = side3;
    }
    public double calculateArea() {
        double s = (side1 + side2 + side3) / 2;
        return Math.sqrt(s * (s - side1) * (s - side2) * (s - side3));
    }
}
```

OUTPUT:

```
C:\Users\admin\Desktop\java>JAVAC AreaCalculator.java

C:\Users\admin\Desktop\java>JAVA AreaCalculator
Area of Circle: 78.53981633974483
Area of Rectangle: 24.0
Area of Trapezoid: 54.0
Area of Triangle: 6.0
```

2. Define a class called Rectangle with member variables length and width. Use appropriate member functions to calculate the perimeter and area of the rectangle. Define another member function int sameArea(Rectangle) that has one parameter of type Rectangle. sameArea returns 1 if the two Rectangles have the same area, and returns 0 if they dont. Use appropriate constructors to initialize the member variables(Use both default and parameterized constructor)

CODE:

```
public class Rectangle {
   private double length;
   private double width;
   public Rectangle() {
      length = 0;
      width = 0;
   }
   public Rectangle(double length, double width) {
      this.length = length;
      this.width = width;
   }
   public double calculatePerimeter() {
      return 2 * (length + width);
   }
   public double calculateArea()
      {return length * width;
   }
   public int sameArea(Rectangle other) {
      double area1 = this.calculateArea();
      double area2 = other.calculateArea();
      if (area1 == area2) {
         return 1;
      } else {
         return 0;
      }
   }
   public double getLength()
      {return length;
   }

   public void setLength(double length)
      {this.length = length;
   }

   public double getWidth()
      {return width;
```

```
    }

    public void setWidth(double width)
      {this.width = width;
    }
    public static void main(String[] args) {
      Rectangle rectangle1 = new Rectangle(5, 4);
      Rectangle rectangle2 = new Rectangle(4, 5);
      System.out.println("Perimeter of rectangle1: " + rectangle1.calculatePerimeter());
      System.out.println("Area of rectangle1: " + rectangle1.calculateArea());

      System.out.println("Perimeter of rectangle2: " + rectangle2.calculatePerimeter());
      System.out.println("Area of rectangle2: " + rectangle2.calculateArea());

      System.out.println("Are the areas of rectangle1 and rectangle2 the same? " +
rectangle1.sameArea(rectangle2));
    }
  }
```

OUTPUT:

```
C:\Users\admin\Desktop\java>javac Rectangle.java

C:\Users\admin\Desktop\java>java Rectangle
Perimeter of rectangle1: 18.0
Area of rectangle1: 20.0
Perimeter of rectangle2: 18.0
Area of rectangle2: 20.0
Are the areas of rectangle1 and rectangle2 the same? 1
```

3. Write a main function to create two rectangle objects and display its area and perimeter. Check whether the two Rectangles have the same area and print a message indicating the result. (Use the concept of this pointer too)

CODE:

```
public class Rectangle {
   private double length;
   private double width;
   public Rectangle() {
      length = 0;
      width = 0;
   }
   public Rectangle(double length, double width) {
      this.length = length;
      this.width = width;
   }
   public double calculatePerimeter() {
      return 2 * (length + width);
   }
   public double calculateArea()
      {return length * width;
   }
   public int sameArea(Rectangle other) {
      double area1 = this.calculateArea();
      double area2 = other.calculateArea();
      if (area1 == area2) {
         return 1;
      } else {
         return 0;
      }
   }
   public double getLength()
      {return length;
   }
   public void setLength(double length)
      {this.length = length;
   }
   public double getWidth()
      {return width;
   }
   public void setWidth(double width)
      {this.width = width;
   }
   public static void main(String[] args) {
```

```
        Rectangle rectangle1 = new Rectangle(5, 4);
        Rectangle rectangle2 = new Rectangle(4, 5);
        System.out.println("Rectangle 1:");
        System.out.println("Area: " + rectangle1.calculateArea());
        System.out.println("Perimeter: " + rectangle1.calculatePerimeter());
        System.out.println("\nRectangle 2:");
        System.out.println("Area: " + rectangle2.calculateArea());
        System.out.println("Perimeter: " + rectangle2.calculatePerimeter());

        if (rectangle1.sameArea(rectangle2) == 1) {
           System.out.println("\nBoth rectangles have the same area.");
        } else {
           System.out.println("\nThe rectangles don't have the same area.");
        }
    }
}
```

OUTPUT:

```
C:\Users\admin\Desktop\java>javac Rectangle.java

C:\Users\admin\Desktop\java>java Rectangle
Rectangle 1:
Area: 20.0
Perimeter: 18.0

Rectangle 2:
Area: 20.0
Perimeter: 18.0

Both rectangles have the same area.
```

4. Write the definition for a class called Complex that has floating point data members for storing real and imaginary parts. Define a function Complex sum(Complex) to add two complex numbers

CODE:

```
public class Complex {
   private double real;
   private double imaginary;
   public Complex() {
      this.real = 0;
      this.imaginary = 0;
   }
   public Complex(double real, double imaginary)
      {this.real = real;
      this.imaginary = imaginary;
   }
   public Complex sum(Complex other) {
      double realSum = this.real + other.real;
      double imaginarySum = this.imaginary +
      other.imaginary;return new Complex(realSum,
      imaginarySum);
   }
   public double getReal()
      {return real;
   }
   public void setReal(double real)
      {this.real = real;
   }
   public double getImaginary()
      {return imaginary;
   }
   public void setImaginary(double imaginary)
      {this.imaginary = imaginary;
   }
   public void display() {
      System.out.println("Complex Number: " + real + " + " + imaginary + "i");
   }
   public static void main(String[] args) {
      Complex num1 = new Complex(2,
      3);Complex num2 = new
      Complex(1, 4);Complex sum =
      num1.sum(num2);
      System.out.print("Sum: ");
      sum.display();
   }
}
```

OUTPUT:

```
C:\Users\admin\Desktop\java>javac Complex.java

C:\Users\admin\Desktop\java>java Complex
Sum: Complex Number: 3.0 + 7.0i
```

5. & return complex number. Write main function to create three complex number objects. Set the value in two objects and call sum() to calculate sum and assign it in third object. Display all complex numbers. (Use the concept of this pointer too.)

CODE:

```
class Complex {
   private float real;
   private float imaginary;

   Complex(float real, float imaginary)
      {this.real = real;
      this.imaginary = imaginary;
   }

   Complex sum(Complex other) {
      float sumReal = this.real + other.real;
      float sumImaginary = this.imaginary +
      other.imaginary;return new Complex(sumReal,
      sumImaginary);
   }

   void display() {
      System.out.println(real + " + " + imaginary + "i");
   }
}

public class TwoFive {
   public static void main(String[] args) {
      // Create three complex number objects
      Complex c1 = new Complex(2.5f, 3.7f);
      Complex c2 = new Complex(1.2f, -4.1f);
      Complex c3;

      System.out.println("First complex number:");
      c1.display();
      System.out.println("Second complex number:");
      c2.display();

      // Call sum() to calculate the sum and assign it to c3
      c3 = c1.sum(c2);

      System.out.println("Sum of the two complex numbers:");
      c3.display();
   }
}
```

OUTPUT:

```
First complex number:
2.5 + 3.7i
Second complex number:
1.2 + -4.1i
Sum of the two complex numbers:
3.7 + -0.39999986i
```

6. Define a class called Time that has hours and minutes as integer. The class has the following member function: *Time sum(Time)* to sum two time object & return time a. Usethe concept of constructor overloading to initialize the time

1.1 Write the definitions for each of the above member functions.

1.2 Write main function to create three time objects. Set the value in two objectsand call sum() to calculate sum and assign it in third object. Display all time objects. (Use the concept of *this* pointer too)

CODE:

```
public class TwoSix {
   public static void main(String[] args)
      {Time t1 = new Time(2, 30);
      Time t2 = new Time(1,
      45);Time t3 = t1.sum(t2);

        System.out.println("Time 1: " + t1.getHours() + " hours " + t1.getMinutes() + "
minutes");
        System.out.println("Time 2: " + t2.getHours() + " hours " + t2.getMinutes() + "
minutes");
      System.out.println("Sum: " + t3.getHours() + " hours " + t3.getMinutes() + " minutes");
   }
}

class Time {
   private int hours;
   private int minutes;

   public Time(int hours, int minutes)
      {this.hours = hours;
      this.minutes = minutes;
   }

   public Time sum(Time t) {
      int totalMinutes = this.hours * 60 + this.minutes + t.hours * 60 + t.minutes;
      int newHours = totalMinutes / 60;
      int newMinutes = totalMinutes % 60;
      return new Time(newHours, newMinutes);
   }

   public int getHours()
      {return hours;
   }
```

```
    public int getMinutes()
       {return minutes;
    }
}
```

OUTPUT:

```
Time 1: 2 hours 30 minutes
Time 2: 1 hours 45 minutes
Sum: 4 hours 15 minutes
```

7. Create a class 'Account' with two overloaded constructors. The first constructor is used for initializing the name of account holder, the account number and the initial amount in the account. The second constructor is used for initializing the name of the account holder, the account number, the addresses, the type of account and the current balance. The Account class is having methods Deposit (), Withdraw (), and Get_Balance(). Make the necessary assumption for data members and return types of the methods. Create objects of Account class and use them.

CODE:

```
import java.util.Scanner;
class Account
{
   private String accountHolderName;
   private int accountNumber;
   private String address;
   private String accountType;
   private double balance;

   public Account(String accountHolderName, int accountNumber, double initialAmount)
{
      this.accountHolderName = accountHolderName;
      this.accountNumber = accountNumber;
      this.balance = initialAmount;
   }
   public Account(String accountHolderName, int accountNumber, String address, String
accountType, double balance)
  {
      this.accountHolderName = accountHolderName;
      this.accountNumber = accountNumber;
      this.address = address;
      this.accountType = accountType;


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

   public void withdraw(double amount) {
      if (amount > balance) {
         System.out.println("Insufficient balance. Withdrawal failed.");
      }
```

```
    else
    {
        balance -= amount;
        System.out.println("Amount withdrawn successfully.");
    }
}


public double getBalance()
{
    return balance;
}
public static void main(String[] args)
{
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter account holder's name:");
    String name1 = scanner.nextLine();
    System.out.println("Enter account number:");
    int number1 = scanner.nextInt();
    System.out.println("Enter initial amount:");
    double initialAmount = scanner.nextDouble();
    scanner.nextLine();
    Account account1 = new Account(name1, number1, initialAmount);
    System.out.println("Enter account holder's name:");
    String name2 = scanner.nextLine();
    System.out.println("Enter account number:");
    int number2 = scanner.nextInt();
    scanner.nextLine();
    System.out.println("Enter address:");
    String address = scanner.nextLine();
    System.out.println("Enter account type:");
    String type = scanner.nextLine();
    System.out.println("Enter current balance:");
    double balance = scanner.nextDouble();
    scanner.nextLine();
    Account account2 = new Account(name2, number2, address, type, balance);
    System.out.println("Initial balances:");
    System.out.println("Account 1 balance: " + account1.getBalance());
    System.out.println("Account 2 balance: " + account2.getBalance());

    System.out.println("Enter amount to deposit into Account 1:");
    double depositAmount = scanner.nextDouble();
    account1.deposit(depositAmount);
```

```
        System.out.println("Enter amount to withdraw from Account 2:");
        double withdrawAmount = scanner.nextDouble();
        account2.withdraw(withdrawAmount);

        System.out.println("Updated balances:");
        System.out.println("Account 1 balance: " + account1.getBalance());
        System.out.println("Account 2 balance: " + account2.getBalance());
        scanner.close();
    }
}
```

OUTPUT:

```
C:\JAVA>java Account
Enter account holder's name:
Emira Liz
Enter account number:
101
Enter initial amount:
50000
Enter account holder's name:
Enola Grace
Enter account number:
102
Enter address:
Nazareth, Fort Kochi
Enter account type:
Savings
Enter current balance:
40000
Initial balances:
Account 1 balance: 50000.0
Account 2 balance: 40000.0
Enter amount to deposit into Account 1:
5000
Amount deposited successfully.
Enter amount to withdraw from Account 2:
3500
Amount withdrawn successfully.
Updated balances:
Account 1 balance: 55000.0
Account 2 balance: 36500.0
```

CYCLE 03:	**Date: 28-01-2024**
Inheritance, method overloading and overriding, Polymorphism	

1. Write a Java program which creates a class named 'Employee' having the following members: Name, Age, Phone number, Address, Salary. It also has a method named 'print-Salary( )' which prints the salary of the Employee. Two classes 'Officer' and 'Manager' inherits the 'Employee' class. The 'Officer' and 'Manager' classes have data members 'specialization' and 'department' respectively. Now, assign name, age, phone number, address and salary to an officer and a manager by making an object of both of these classes and print the same.

CODE:

```
class Employee
  {String name;
  int age;
  String phoneNumber;
  String address;
  double salary;

  public void printSalary() {
    System.out.println("Salary: " + salary);
  }
}
class Officer extends Employee
  {String specialization;
   public Officer(String name, int age, String phoneNumber, String address, double
salary, String specialization) {
    this.name = name;
    this.age = age;
    this.phoneNumber = phoneNumber;
    this.address = address;
    this.salary = salary;
    this.specialization =
    specialization;
  }
  public void displayDetails() {
    System.out.println("Name: " + name);
    System.out.println("Age: " + age);
    System.out.println("Phone Number: " + phoneNumber);
```

```
        System.out.println("Address: " + address);
        printSalary();
        System.out.println("Specialization: " + specialization);
    }
}

class Manager extends Employee {
    String department;

    public Manager(String name, int age, String phoneNumber, String address, double
salary, String department) {
        this.name = name;
        this.age = age;
        this.phoneNumber = phoneNumber;
        this.address = address;
        this.salary = salary;
        this.department = department;
    }

    public void displayDetails() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Phone Number: " + phoneNumber);
        System.out.println("Address: " + address);
        printSalary();
        System.out.println("Department: " + department);
    }
}

public class ThreeOne {
    public static void main(String[] args) {
        Officer officer = new Officer("John Doe", 30, "1234567890", "123 Main St,
Anytown USA", 50000.0, "Software Engineering");
        Manager manager = new Manager("Jane Smith", 35, "9876543210", "456 Oak
Rd, Sometown USA", 70000.0, "IT");

        System.out.println("Officer
        Details:");officer.displayDetails();

        System.out.println("\nManager Details:");
        manager.displayDetails();
    }
}
```

OUTPUT:

```
Officer Details:
Name: John Doe
Age: 30
Phone Number: 1234567890
Address: 123 Main St, Anytown USA
Salary: 50000.0
Specialization: Software Engineering

Manager Details:
Name: Jane Smith
Age: 35
Phone Number: 9876543210
Address: 456 Oak Rd, Sometown USA
Salary: 70000.0
Department: IT
```

2. Write two Java classes Employee and Engineer. Engineer should inherit from Employee class. Employee class to have two methods display() and calcSalary(). Write a program to display the engineer salary and to display from Employee class using a single object instantiation (i.e., only one object creation is allowed).

a. display() only prints the name of the class and does not return any value. Ex. "Name of class is Employee."

b. calcSalary() in Employee displays "Salary of employee is 10000" and calcSalary()in Engineer displays "Salary of employee is 20000."

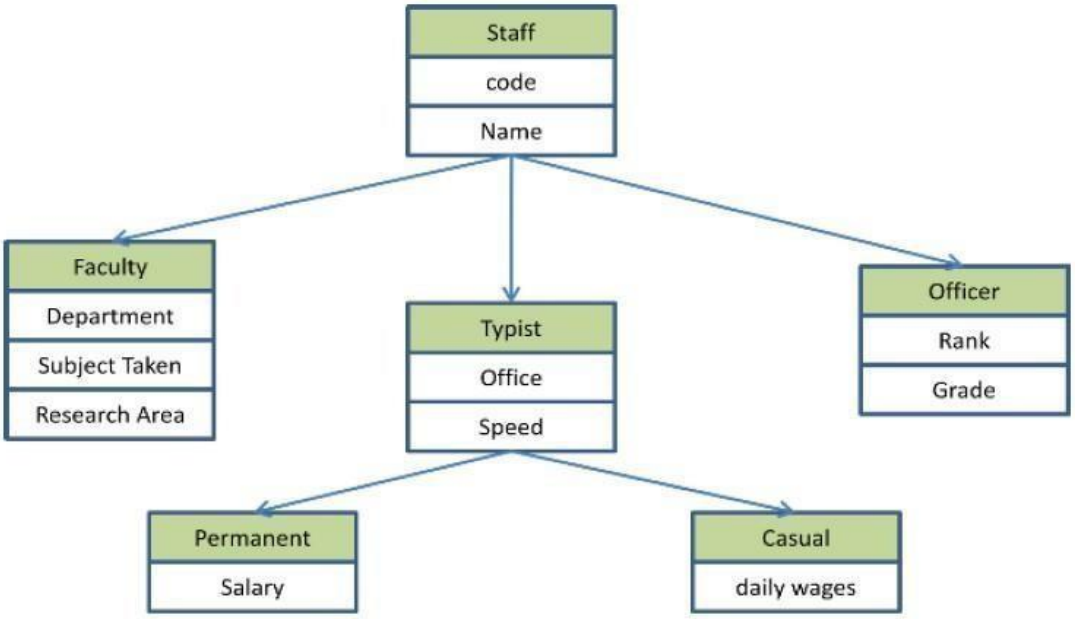
CODE:

```
public class ThreeTwo {
    public static void main(String[] args) {
        Employee employee = new Engineer();
        employee.display();
        employee.calcSalary();
    }
}

class Employee {
    public void display() {
        System.out.println("Name of class is Employee.");
    }

    public void calcSalary() {
        System.out.println("Salary of employee is 10000.");
    }
}

class Engineer extends Employee
    {@Override
    public void calcSalary() {
        System.out.println("Salary of employee is 20000.");
    }
}
```

OUTPUT:

```
Name of class is Employee.
Salary of employee is 20000.
```

3. Write a Java program to implement the following level of inheritance.



CODE:

```
class Staff {
   String code;
   String name;

   Staff(String code, String name)
      {this.code = code;
      this.name = name;
   }
}
class Typist extends Staff
   {String office;
   int speed;

   Typist(String code, String name, String office, int speed)
      {super(code, name);
      this.office = office;
      this.speed = speed;
   }
}
class Faculty extends Staff
   {String   department;
   String subjectTaken;
   String researchArea;
```

```
    Faculty(String code, String name, String department, String subjectTaken, String
researchArea) {
      super(code, name);
      this.department = department;
      this.subjectTaken = subjectTaken;
      this.researchArea = researchArea;
   }
}
class Officer extends Staff
   {String rank;
   String grade;

   Officer(String code, String name, String rank, String grade)
      {super(code, name);
      this.rank = rank;
      this.grade = grade;
   }
}
class Permanent extends Faculty {
   double salary;

     Permanent(String code, String name, String department, String subjectTaken, String
researchArea, double salary) {
      super(code, name, department, subjectTaken, researchArea);
      this.salary = salary;
   }
}
class Casual extends Faculty
   {double dailyWages;

    Casual(String code, String name, String department, String subjectTaken, String
researchArea, double dailyWages) {
      super(code, name, department, subjectTaken, researchArea);
      this.dailyWages = dailyWages;
   }
}

public class TwoThree {
   public static void main(String[] args) {
      Typist typist = new Typist("T001", "John Doe", "Administration", 60);
       Permanent permanent = new Permanent("F001", "Jane Smith", "Computer Science",
"Data Structures", "Algorithms", 50000.0);
         Casual casual = new Casual("F002", "Bob Johnson", "Mathematics", "Calculus",
"Analysis", 500.0);
```

```
    Officer officer = new Officer("O001", "Alice Williams", "Lieutenant", "Grade
    3");System.out.println("Typist: " + typist.name + ", Office: " + typist.office);
        System.out.println("Permanent Faculty: " + permanent.name + ", Salary: " +
permanent.salary);
        System.out.println("Casual Faculty: " + casual.name + ", Daily Wages: " +
casual.dailyWages);
    System.out.println("Officer: " + officer.name + ", Rank: " + officer.rank);
  }
}
```

OUTPUT:

```
Typist: John Doe, Office: Administration
Permanent Faculty: Jane Smith, Salary: 50000.0
Casual Faculty: Bob Johnson, Daily Wages: 500.0
Officer: Alice Williams, Rank: Lieutenant
```

4. Write a java program to create an abstract class named Shape that contains an empty method named numberOfSides(). Provide three classes named Rectangle, Triangleand Hexagon such that each one of the classes extends the class Shape. Each one of the classes contains only the method numberOfSides( ) that shows the number of sides in thegiven geometrical structures.

CODE:

```
abstract class Shape {
   abstract int numberOfSides();
}
class Rectangle extends Shape
   {int numberOfSides() {
      return 4;
   }
}
class Triangle extends Shape
   {int numberOfSides() {
      return 3;
   }
}
class Hexagon extends Shape
   {int numberOfSides() {
      return 6;
   }
}
public class ThreeFour {
   public static void main(String[] args) {
      Rectangle rectangle = new Rectangle();
      Triangle triangle = new Triangle();
      Hexagon hexagon = new Hexagon();

            System.out.println("Number of sides in a Rectangle: "
                                    +rectangle.numberOfSides());
      System.out.println("Number of sides in a Triangle: " +
            triangle.numberOfSides());System.out.println("Number of
                                    sidesin a Hexagon: "
                                    +

hexagon.numberOfSides());
   }
}
```

OUTPUT:

```
Number of sides in a Rectangle: 4
Number of sides in a Triangle: 3
Number of sides in a Hexagon: 6
```

5. write a program to represent geometric shapes and some operations that can be performed on them. The idea here is that shapes in higher dimensions inherit data from lower dimensional shapes. For example a cube is a three dimensional square. A sphere is a three dimensional circle and a glome is a four dimensional circle. A cylinder is another kind of three dimensional circle. The circle, sphere, cylinder, and glome all share the attribute radius. The square and cube share the attribute side length. There are various ways to use inheritance to relate these shapes but please follow the inheritance described in the table below.

All shapes inherit getName() from the superclass Shape.

Specification:

Your program will consist of the following classes: Shape, Circle, Square,Cube, Sphere, Cylinder, and Glome and two interfaces Area and Volume Your classes may only have the class variable specified in the table belowand the methods defined in the two interfaces Area and Volume. You willimplement the methods specified in the Area and Volume interfaces andhave them return the appropriate value for each shape. Class Shape willhave a single public method called getName that returns a string.

Class	Class Variable	Constructor	Extends	Implements
Shape	String name	Shape()		
Circle	double radius	Circle(double r, String n)	Shape	Area
Square	double side	Square(double s, String n)	Shape	Area
Cylinder	double height	Cylinder(double h, double r, String n)	Circle	Volume
Sphere	None	Sphere(double r, String n)	Circle	Volume
Cube	None	Cube(double s, String n)	Square	Volume
Glome	None	Glome(double r, String n)	Sphere	Volume

Note: the volume of a glome is $0.5(\pi^2)r^4$ where r is the radius.

CODE:

```
interface Area {
   double getArea();
}
interface Volume {
   double getVolume();
}
class Shape {
   String name;
   Shape(String name) {
```

```
        this.name = name;
    }
    String getName()
        {return name;
    }
}
class Circle extends Shape implements Area
    {double radius;
    Circle(double radius, String name)
        {super(name);
        this.radius = radius;
    }
    public double getArea() {
        return Math.PI * radius * radius;
    }
}
class Square extends Shape implements Area {
    double side;
    Square(double side, String name) {
        super(name);
        this.side = side;
    }
    public double getArea()
        {return side * side;
    }
}
class Cylinder extends Circle implements Volume
    {double height;
    Cylinder(double height, double radius, String name)
        {super(radius, name);
        this.height = height;
    }
    public double getVolume() {
        return Math.PI * radius * radius * height;
    }
}
class Sphere extends Circle implements Volume
    {Sphere(double radius, String name) {
        super(radius, name);
    }
    public double getVolume() {
        return (4.0 / 3.0) * Math.PI * radius * radius * radius;
    }
}
```

```
class Cube extends Square implements Volume {
   Cube(double side, String name) {
      super(side, name);
   }
   public double getVolume()
      {return side * side * side;
   }
}
class Glome extends Sphere implements Volume {
   Glome(double radius, String name) {
      super(radius, name);
   }
   public double getVolume() {
      return 0.5 * Math.PI * Math.PI * radius * radius * radius * radius;
   }
}
public class ThreeFive {
   public static void main(String[] args) {
      Circle circle = new Circle(5.0,
      "Circle");
      Square square = new Square(4.0, "Square");
      Cylinder cylinder = new Cylinder(10.0, 3.0,
      "Cylinder");Sphere sphere = new Sphere(6.0,
      "Sphere");
      Cube cube = new Cube(7.0, "Cube");
      Glome glome = new Glome(2.0,
      "Glome");
      System.out.println("Shape: " + circle.getName() + ", Area: " + circle.getArea());
      System.out.println("Shape: " + square.getName() + ", Area: " + square.getArea());
            System.out.println("Shape:    "   +   cylinder.getName()    +   ",   Volume:    "
                                          +cylinder.getVolume());
      System.out.println("Shape: " + sphere.getName() + ", Volume: " + sphere.getVolume());
      System.out.println("Shape: " + cube.getName() + ", Volume: " + cube.getVolume());
      System.out.println("Shape: " + glome.getName() + ", Volume: " + glome.getVolume());
   }
}
```

OUTPUT:

```
Shape: Circle, Area: 78.53981633974483
Shape: Square, Area: 16.0
Shape: Cylinder, Volume: 282.7433388230814
Shape: Sphere, Volume: 904.7786842338604
Shape: Cube, Volume: 343.0
Shape: Glome, Volume: 78.95683520871486
```

6. Define an interface "Operations" which has method area(), volume(). Define a constant PI having value 3.14. Create class a Cylinder(with member variable height) whichimplements this interface. Create one object and calculate area and volume. Add RequiredConstructors.

CODE:

```
interface Operations {
   double PI = 3.14;
   double area();
   double volume();
}

class Cylinder implements Operations
   {private double radius;
   private double height;
   Cylinder(double radius, double height)
      {this.radius = radius;
      this.height = height;
   }
   public double area() {
      return 2 * PI * radius * (radius + height);
   }
   public double volume() {
      return PI * radius * radius * height;
   }
}

public class ThreeSix {
   public static void main(String[] args) {
      Cylinder cylinder = new Cylinder(5.0,
      10.0);double area = cylinder.area();
      System.out.println("Area of the cylinder: " + area);
      double volume = cylinder.volume();
      System.out.println("Volume of the cylinder: " + volume);
   }
}
```

OUTPUT:

```
Area of the cylinder: 471.00000000000006
Volume of the cylinder: 785.0
```

7. Write a program that illustrates interface inheritance. Interface P is extended by P1and P2. Interface P12 inherits from both P1 and P2.Each interface declares one constant and one method. class Q implements P12.Instantiate Q and invoke each of its methods. Each method displays one of the constants.

CODE:

```
interface P {
   int CONSTANT_P = 10;
   void methodP();
}
interface P1 extends P {
   int CONSTANT_P1 =
   20;
   void methodP1();
}
interface P2 extends P {
   int CONSTANT_P2 =
   30;
   void methodP2();
}
interface P12 extends P1, P2 {
}
class Q implements P12 {
   public void methodP()
   {
      System.out.println("Constant from P: " + CONSTANT_P);
   }
   public void methodP1() {
      System.out.println("Constant from P1: " + CONSTANT_P1);
   }
   public void methodP2() {
      System.out.println("Constant from P2: " + CONSTANT_P2);
   }
}
public class ThreeSeven {
   public static void main(String[] args)
      {Q q = new Q();
      q.methodP();
      q.methodP1();
      q.methodP2();
   }
}
```

OUTPUT:

```
Constant from P: 10
Constant from P1: 20
Constant from P2: 30
```

CYCLE 04:	**Date: 5-02-2024**
Multithreading	

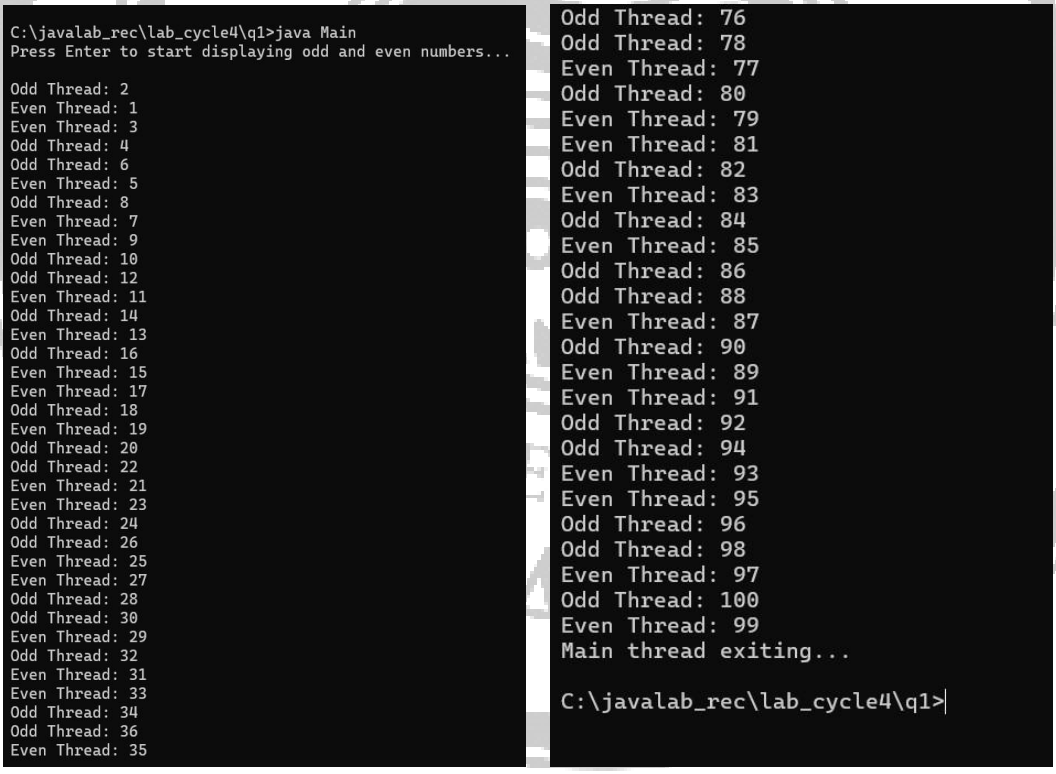
1. Write a Java program to create two threads: One for displaying all odd number between 1 and 100 and second thread for displaying all even numbers between 1 and 100.Create a multithreaded program by creating a subclass of Thread and then creating, initializing, and starting two Thread objects from your class. The threads will execute concurrently Main thread should wait until all the other thread terminates its execution(using join()).

CODE:

```
import java.util.Scanner;

class OddThread extends Thread {
    public void run() {
        for (int i = 1; i <= 100; i += 2) {
            System.out.println("Odd Thread: " + i);
            try {
                Thread.sleep(100); // Just for demonstration purposes
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

class EvenThread extends Thread {
    public void run() {
        for (int i = 2; i <= 100; i += 2) {
            System.out.println("Even Thread: " + i);
            try {
                Thread.sleep(100); // Just for demonstration purposes
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Press Enter to start displaying odd and even numbers...");
        scanner.nextLine();
```

```
        OddThread oddThread = new OddThread();
        EvenThread evenThread = new EvenThread();

        oddThread.start();
        evenThread.start();

        try {
            oddThread.join();
            evenThread.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        System.out.println("Main thread exiting...");
    }
}
```

OUTPUT :



```
C:\javalab_rec\lab_cycle4\q1>java Main
Press Enter to start displaying odd and even numbers...

Odd Thread: 2
Even Thread: 1
Even Thread: 3
Odd Thread: 4
Odd Thread: 6
Even Thread: 5
Odd Thread: 8
Even Thread: 7
Even Thread: 9
Odd Thread: 10
Odd Thread: 12
Even Thread: 11
Odd Thread: 14
Even Thread: 13
Odd Thread: 16
Even Thread: 15
Even Thread: 17
Odd Thread: 18
Even Thread: 19
Odd Thread: 20
Odd Thread: 22
Even Thread: 21
Even Thread: 23
Odd Thread: 24
Odd Thread: 26
Even Thread: 25
Even Thread: 27
Odd Thread: 28
Odd Thread: 30
Even Thread: 29
Odd Thread: 32
Even Thread: 31
Even Thread: 33
Odd Thread: 34
Odd Thread: 36
Even Thread: 35
Odd Thread: 76
Odd Thread: 78
Even Thread: 77
Odd Thread: 80
Even Thread: 79
Even Thread: 81
Odd Thread: 82
Even Thread: 83
Odd Thread: 84
Even Thread: 85
Odd Thread: 86
Odd Thread: 88
Even Thread: 87
Odd Thread: 90
Even Thread: 89
Even Thread: 91
Odd Thread: 92
Odd Thread: 94
Even Thread: 93
Even Thread: 95
Odd Thread: 96
Odd Thread: 98
Even Thread: 97
Odd Thread: 100
Even Thread: 99
Main thread exiting...

C:\javalab_rec\lab_cycle4\q1>
```

2. Write a Java program that set thread priorities and display the priority.

CODE:

```
import java.lang.Thread;
import java.util.Scanner;

class PriorityThread extends Thread {
    public PriorityThread(String name) {
        super(name);
    }

    public void run() {
        System.out.println("Thread Name: " + Thread.currentThread().getName() +
                ", Priority: " + Thread.currentThread().getPriority());
    }
}

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

        // Take user input for priorities
        System.out.print("Enter priority for Thread 1 (1-10): ");
        int priority1 = scanner.nextInt();

        System.out.print("Enter priority for Thread 2 (1-10): ");
        int priority2 = scanner.nextInt();

        System.out.print("Enter priority for Thread 3 (1-10): ");
        int priority3 = scanner.nextInt();

        PriorityThread thread1 = new PriorityThread("Thread 1");
        PriorityThread thread2 = new PriorityThread("Thread 2");
        PriorityThread thread3 = new PriorityThread("Thread 3");


        // Set priorities
        thread1.setPriority(priority1);
        thread2.setPriority(priority2);
        thread3.setPriority(priority3);
```

```
        // Start threads
        thread1.start();
        thread2.start();
        thread3.start();

        scanner.close();
    }
}
```

OUTPUT:

```
C:\javalab_rec\lab_cycle4\q2>javac Main.java

C:\javalab_rec\lab_cycle4\q2>java Main
Enter priority for Thread 1 (1-10): 3
Enter priority for Thread 2 (1-10): 8
Enter priority for Thread 3 (1-10): 2
Thread Name: Thread 2, Priority: 8
Thread Name: Thread 1, Priority: 3
Thread Name: Thread 3, Priority: 2
```

3. Write a java program that implements a multi-thread application that has three threads. The first thread generates random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number

CODE:

```
import java.util.Random;
class RandomNumberGenerator extends Thread {
   private boolean running;
   public RandomNumberGenerator() {
      this.running = true;
   }
   public void stopGenerating() {
      this.running = false;
   }
  @Override
   public void run() {
      Random random = new Random();
      while (running) {
         try {
            Thread.sleep(1000);
            int number = random.nextInt(100);
            System.out.println("Generated number: " + number);
            if (number % 2 == 0) { // Check if the number is even
               SquareThread squareThread = new SquareThread(number);
               squareThread.start(); // Start SquareThread
            }
            else {
               CubeThread cubeThread = new CubeThread(number);
               cubeThread.start();
            }
         } catch (InterruptedException e) {
            e.printStackTrace();
         }
      }
   }
}
class SquareThread extends Thread {
   private int number;
   public SquareThread(int number) {
      this.number = number;
   }
   @Override
   public void run() {
      System.out.println("Square of " + number + ": " + (number * number));
   }
```

```
}
class CubeThread extends Thread {
   private int number;
   public CubeThread(int number) {
      this.number = number;
   }
   @Override
   public void run() {
      System.out.println("Cube of " + number + ": " + (number * number * number));
   }
}
public class Main {
   public static void main(String[] args) {

RandomNumberGenerator randomNumberGenerator = new RandomNumberGenerator();
      randomNumberGenerator.start();
      try {
         Thread.sleep(10000);
      } catch (InterruptedException e) {
         e.printStackTrace();
      }
      randomNumberGenerator.stopGenerating();
   }
}
```

OUTPUT :

```
C:\javalab_rec\lab_cycle4\q3>javac Main.java

C:\javalab_rec\lab_cycle4\q3>java Main
Generated number: 53
Cube of 53: 148877
Generated number: 15
Cube of 15: 3375
Generated number: 0
Square of 0: 0
Generated number: 5
Cube of 5: 125
Generated number: 17
Cube of 17: 4913
Generated number: 24
Square of 24: 576
Generated number: 61
Cube of 61: 226981
Generated number: 73
Cube of 73: 389017
Generated number: 51
Cube of 51: 132651
Generated number: 15
Cube of 15: 3375
```

4) Write a program to illustrate creation of threads using runnable interface. (start method start each of the newly created thread. Inside the run method there is sleep() for suspend the thread for 500 milliseconds). Main thread should wait until all the other thread terminates its execution (using join()).

CODE:

```
import java.util.Scanner;

class MyRunnable implements Runnable {
    public void run() {
        try {
            System.out.println(Thread.currentThread().getName() + " is running.");
            Thread.sleep(500);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of threads: ");
        int numThreads = scanner.nextInt();
        scanner.close();
        Thread[] threads = new Thread[numThreads];
        for (int i = 0; i < numThreads; i++) {
            threads[i] = new Thread(new MyRunnable());
            threads[i].start();
        }
        for (Thread thread : threads) {
            try {
                thread.join();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        System.out.println("All threads have terminated.");
    }
}
```

OUTPUT :

```
C:\javalab_rec\lab_cycle4\q4>javac Main.java

C:\javalab_rec\lab_cycle4\q4>java Main
Enter the number of threads: 4
Thread-0 is running.
Thread-1 is running.
Thread-3 is running.
Thread-2 is running.
All threads have terminated.
```

5) Write a java program showing a typical invocation of banking operations via multiple threads. Create three threads and 2 methods deposit and withdraw methods to add the amount to the account and withdraw an amount from the account respectively. As the threads concurrently run the method, avoid the unpredictable behavior. (Use synchronization).

CODE:

```
import java.util.Scanner;

class BankAccount {
   private int balance;
   public BankAccount(int initialBalance) {
      balance = initialBalance;
   }
   public synchronized void deposit(int amount) {
      System.out.println(Thread.currentThread().getName() + " is depositing $" + amount);
      balance += amount;
      System.out.println("New balance after deposit by " + Thread.currentThread().getName()
+ ": $" + balance);
   }
   public synchronized void withdraw(int amount) {
      System.out.println(Thread.currentThread().getName() + " is withdrawing $" + amount);
      if (balance >= amount) {
         balance -= amount;
         System.out.println("New balance after withdrawal by " +
Thread.currentThread().getName() + ": $" + balance);
      } else {
         System.out.println("Insufficient balance for withdrawal by " +
Thread.currentThread().getName());
      }
   }
}

public class Main {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter initial balance:  ");
      int initialBalance = scanner.nextInt();
      BankAccount account = new BankAccount(initialBalance);
      System.out.print("Enter deposit amount: ");
      int depositAmount = scanner.nextInt();
      System.out.print("Enter withdrawal amount: ");
      int withdrawalAmount = scanner.nextInt();
      scanner.close();
```

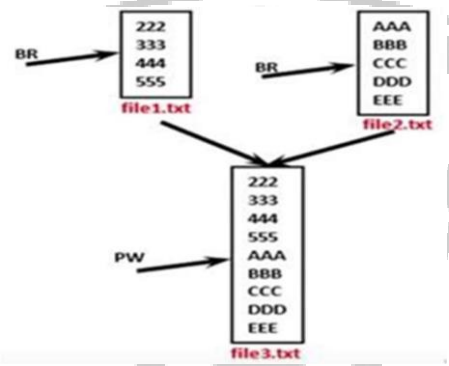
```
        Thread thread1 = new Thread(() -> {
            account.deposit(depositAmount);
        }, "Thread-1");

        Thread thread2 = new Thread(() -> {
            account.withdraw(withdrawalAmount);
        }, "Thread-2");

        Thread thread3 = new Thread(() -> {
            account.deposit(depositAmount);
        }, "Thread-3");
        thread1.start();
        thread2.start();
        thread3.start();
    }
}
```

OUTPUT:

```
C:\javalab_rec\lab_cycle4\q5>javac Main.java

C:\javalab_rec\lab_cycle4\q5>java Main
Enter initial balance: 3000
Enter deposit amount: 1000
Enter withdrawal amount: 200
Thread-1 is depositing $1000
New balance after deposit by Thread-1: $4000
Thread-3 is depositing $1000
New balance after deposit by Thread-3: $5000
Thread-2 is withdrawing $200
New balance after withdrawal by Thread-2: $4800
```

CYCLE 05:	Date: 15-02-2024
Input-Output, File Management and exception handling	

1. **Write a Java Program to merge data from two files into a third file. (Handle all file related exceptions)**



CODE:

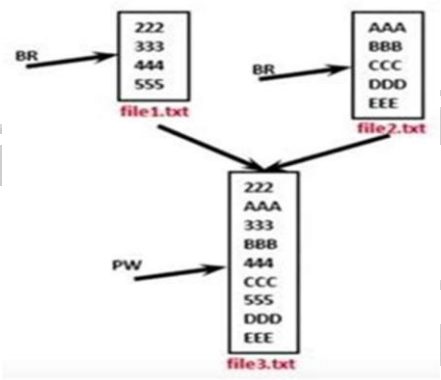
```
import java.io.*;
import java.util.*;

public class FTwoCopy {
   public static void main(String[] args) {
      FileReader fr;
      BufferedReader br;
      FileWriter fwr;
      BufferedWriter bwr;
      Scanner s = new Scanner(System.in);
      String f1,f2,f3;
      try{
         System.out.println("Enter first file name : ");
         f1 = s.nextLine();
         System.out.println("Enter second file name : ");
         f2 = s.nextLine();
         System.out.println("Enter destination file name : ");
         f3 = s.nextLine();
         fr = new FileReader(new File(f1));
         br = new BufferedReader(fr);
         fwr = new FileWriter(new File(f3));
         bwr = new BufferedWriter(fwr);
         String data = br.readLine();
         while(data!=null){
            bwr.write(data);
            bwr.newLine();
            data=br.readLine();
         }
         fr = new FileReader(new File(f2));
         br = new BufferedReader(fr);
         data = br.readLine();
```

```
            while(data!=null){
               bwr.write(data);
               bwr.newLine();
               data=br.readLine();
            }
            br.close();
            bwr.close();
         }
         catch(Exception e){
            System.out.println(e);
         }
      }
```

OUTPUT:

```
C:\Users\cclab25\Desktop\javaa>java FTwoCopy
Enter first file name :
file1.txt
Enter second file name :
file2.txt
Enter destination file name :
f5.txt
```

2. **Write a Java Program to perform file merge operation where merging should be done by line by line alternatively. (Handle all file related exceptions)**



CODE:

```
import java.io.*;
import java.util.*;

public class FTwoCopyAlt {
   public static void main(String[] args) {
      FileReader fr1,fr2;
      BufferedReader br1,br2;
      FileWriter fwr;
      BufferedWriter bwr;
      Scanner s = new Scanner(System.in);
      String f1,f2,f3;
      try{
         System.out.println("Enter first file name : ");
         f1 = s.nextLine();
         System.out.println("Enter second file name : ");
         f2 = s.nextLine();
         System.out.println("Enter destination file name : ");
         f3 = s.nextLine();
         fr1 = new FileReader(new File(f1));
         br1 = new BufferedReader(fr1);
         fr2 = new FileReader(new File(f2));
         br2 = new BufferedReader(fr2);
         fwr = new FileWriter(new File(f3));
         bwr = new BufferedWriter(fwr);


         String data1 = br1.readLine();
         String data2 = br2.readLine();
         while(data1!=null && data2!=null){
            bwr.write(data1);
            bwr.newLine();
            bwr.write(data2);
```

```
            bwr.newLine();
            data1=br1.readLine();
            data2 = br2.readLine();
        }
        while(data1!=null){
            bwr.write(data1);
            bwr.newLine();
            data1=br1.readLine();
        }
        while(data2!=null){
            bwr.write(data1);
            bwr.newLine();
            data2=br2.readLine();

        }
        br1.close();
        br2.close();
        bwr.close();
    }
    catch(FileNotFoundException e1){
        System.out.println("Error "+e1);
    }
    catch(Exception e){
        System.out.println(e);
    }
}
```

OUTPUT:

```
C:\Users\cclab25\Desktop\javaa>javac FTwoCopyAlt.java

C:\Users\cclab25\Desktop\javaa>java FTwoCopyAlt
Enter first file name :
file1.txt
Enter second file name :
file2.txt
Enter destination file name :
f6
```

3. Write a Java program that reads a set of real numbers from a file and displays the minimum, maximum, average, and range of the numbers in the file. The user should be able to enter the name of the input file from the keyboard.

CODE:

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Scanner;

public class NumberStatsFromFile {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter the input file name: ");
    String inputFileName = scanner.nextLine();
    try {
      ArrayList<Double> numbers = readNumbersFromFile(inputFileName);
      if (numbers != null && numbers.size() > 0) {
        double min = Collections.min(numbers);
        double max = Collections.max(numbers);
        double sum = 0;
        for (double num : numbers) {
          sum += num;
        }
        double average = sum / numbers.size();
        double range = max - min;
        System.out.println("Minimum: " + min);
        System.out.println("Maximum: " + max);
        System.out.println("Average: " + average);
        System.out.println("Range: " + range);
      } else {
        System.out.println("No numbers found in the file.");
      }
    } catch (IOException e) {
      System.out.println("An error occurred: " + e.getMessage());
    }
  }
  private static ArrayList<Double> readNumbersFromFile(String fileName) throws
IOException {
    ArrayList<Double> numbers = new ArrayList<>();
    BufferedReader reader = new BufferedReader(new FileReader(fileName));
    String line;
    while ((line = reader.readLine()) != null) {
      try {
```

```
                double number = Double.parseDouble(line);
                numbers.add(number);
            } catch (NumberFormatException e) {
                System.out.println("Warning: Ignoring non-numeric value: " + line);
            }
        }
        reader.close();
        return numbers;
    }
}
```

OUTPUT:

```
C:\Users\hp\Desktop\java>javac  NumberStatsFromFile.java

C:\Users\hp\Desktop\java>java  NumberStatsFromFile
Enter the input file name: realnumbers.txt
Minimum: 2.0
Maximum: 10.2
Average: 6.137499999999999
Range: 8.2
```

4. Write a program that reads the contents of a file and creates an exact copy of the file, except that each line is numbered. For example, if the input file contains the following text:

Two roads diverged in a yellow wood,
And sorry I could not travel both
And be one traveler, long I stood
And looked down one as far as I could
To where it bent in the undergrowth;

then the output file should appear something like this:

1: Two roads diverged in a yellow wood,
2: And sorry I could not travel both
3: And be one traveler, long I stood
4: And looked down one as far as I could
5: To where it bent in the undergrowth;

The user should be able to enter the names of the input and output files from the keyboard.

CODE:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;
public class FileCopyWithLineNumbers {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter the input file name: ");
      String inputFileName = scanner.nextLine();
      System.out.print("Enter the output file name: ");
      String outputFileName = scanner.nextLine();
      try {
         BufferedReader reader = new BufferedReader(new
FileReader(inputFileName));
         BufferedWriter writer = new BufferedWriter(new FileWriter(outputFileName));
         String line;
         int lineNumber = 1;
         while ((line = reader.readLine()) != null) {
            writer.write(lineNumber + ": " + line);
            writer.newLine();
            lineNumber++;
```

```
            reader.close();
            writer.close();

            System.out.println("File copied successfully!");
        } catch (IOException e) {
            System.out.println("An error occurred: " + e.getMessage());
        }
    }
}
```

OUTPUT:

```
C:\Users\hp\Desktop\java>javac FileCopyWithLineNumbers.java

C:\Users\hp\Desktop\java>java FileCopyWithLineNumbers
Enter the input file name: input.txt
Enter the output file name: output.txt
File copied successfully!
```

5. Write a Java program that reads a line of integers, and then displays each integer, and the sum of all the integers. (Use StringTokenizer class of java.util).

CODE:

```
import java.util.Scanner;
import java.util.StringTokenizer;
public class IntegerSum {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter a line of integers separated by spaces: ");
      String input = scanner.nextLine();
      StringTokenizer tokenizer = new StringTokenizer(input);
      int sum = 0;
      System.out.println("Individual Integers:");
      while (tokenizer.hasMoreTokens()) {
         String token = tokenizer.nextToken();
         try {
            int number = Integer.parseInt(token);
            System.out.println(number);
            sum += number;
         } catch (NumberFormatException e) {
            System.out.println("Error: \"" + token + "\" is not a valid integer.");
         }
      }
      System.out.println("Sum of all integers: " + sum);
   }
}
```

OUTPUT:

```
C:\Users\cclab33\Desktop\java>javac IntegerSum.java

C:\Users\cclab33\Desktop\java>java IntegerSum
Enter a line of integers separated by spaces: 2 3 -5 -6 8
Individual Integers:
2
3
-5
-6
8
Sum of all integers: 2
```

6. Write a Java program that displays the number of characters, lines and words in a text file.

CODE:

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
public class FileStats {
   public static void main(String[] args) {
      String filePath ="arya.txt";
      try {
         FileStatistics stats = getFileStatistics(filePath);
         System.out.println("Number of characters: " + stats.getCharacterCount());
         System.out.println("Number of lines: " + stats.getLineCount());
         System.out.println("Number of words: " + stats.getWordCount());
      } catch (IOException e) {
         System.out.println("An error occurred: " + e.getMessage());
      }
   }
   public static FileStatistics getFileStatistics(String filePath) throws IOException {
      int charCount = 0;
      int lineCount = 0;
      int wordCount = 0;
      try (BufferedReader reader = new BufferedReader(new FileReader(filePath))) {
         String line;
         while ((line = reader.readLine()) != null) {
            lineCount++;
            charCount += line.length();
            String[] words = line.split("\\s+");
            wordCount += words.length;
         }
      }
      return new FileStatistics(charCount, lineCount, wordCount);
   }
}
class FileStatistics {
   private final int characterCount;
   private final int lineCount;
   private final int wordCount;
   public FileStatistics(int characterCount, int lineCount, int wordCount) {
      this.characterCount = characterCount;
      this.lineCount = lineCount;
      this.wordCount = wordCount;
   }
   public int getCharacterCount() {
      return characterCount;
```

```
    }
    public int getLineCount() {
        return lineCount;
    }
    public int getWordCount() {
        return wordCount;
    }
}
```

OUTPUT:

```
C:\Users\cclab31\Documents\java Lab>javac FileStats.java

C:\Users\cclab31\Documents\java Lab>java FileStats
Number of characters: 26
Number of lines: 2
Number of words: 7
```

7. Create a class Student with attributes roll no, name, age and course (use user inputs).If age of student is not in between 15 and 21 then generate an exception to handle it.

CODE:

```
import java.util.Scanner;
public class Student {
   private int rollNo;
   private String name;
   private int age;
   private String course;
   public Student(int rollNo, String name, int age, String course) throws
InvalidAgeException {
      if (age < 15 || age > 21) {
         throw new InvalidAgeException("Age must be between 15 and 21.");
      }
      this.rollNo = rollNo;
      this.name = name;
      this.age = age;
      this.course = course;
   }
   public void displayInfo() {
      System.out.println("Roll No: " + rollNo);
      System.out.println("Name: " + name);
      System.out.println("Age: " + age);
      System.out.println("Course: " + course);
   }
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter Roll No: ");
      int rollNo = scanner.nextInt();
      scanner.nextLine(); // Consume newline
      System.out.print("Enter  Name: ");
      String name = scanner.nextLine();
      System.out.print("Enter Age: ");
      int age = scanner.nextInt();
      scanner.nextLine(); // Consume newline
      System.out.print("Enter Course: ");
      String course = scanner.nextLine();
      try {
         Student student = new Student(rollNo, name, age, course);
         student.displayInfo();
      } catch (InvalidAgeException e) {
         System.out.println("Invalid Age: " + e.getMessage());
      }
   }
```

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

OUTPUT:

```
Enter Roll No: 05
Enter Name: joseph
Enter Age: 21
Enter Course: MCA
Roll No: 5
Name: joseph
Age: 21
Course: MCA
```

8. Write a Java program to define a class salesman with the attributes name, salesman code, sales amount and commission(use user inputs). The Company calculates the commission of a salesman according to the following formula:

(i) 8% if sales <2000
(ii) 10% sales if sales>=2000 and but <=5000
(iii) 12% if sales exceeds 5000

Create salesman objects and find the commission of sales. Generate and handle exceptions if sales amount is less than 0.

CODE:

```
import java.util.Scanner;
class Salesman {
    private String name;
    private int salesmanCode;
    private double salesAmount;
    public Salesman(String name, int salesmanCode, double salesAmount) {
        this.name = name;
        this.salesmanCode = salesmanCode;
        this.salesAmount = salesAmount;
    }
    public double calculateCommission() {
        if (salesAmount < 0) {
            throw new IllegalArgumentException("Sales amount cannot be negative.");
        } else if (salesAmount < 2000) {
            return salesAmount * 0.08;
        } else if (salesAmount >= 2000 && salesAmount <= 5000) {
            return salesAmount * 0.10;
        } else {
            return salesAmount * 0.12;
        }
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter salesman name: ");
        String name = scanner.nextLine();
        System.out.print("Enter salesman code: ");
        int code = scanner.nextInt();
        System.out.print("Enter sales amount: ");
        double salesAmount = scanner.nextDouble();
        try {
            Salesman salesman = new Salesman(name, code, salesAmount);
            double commission = salesman.calculateCommission();
            System.out.println("Commission for salesman " + name + " with code " + code
+ " is: $" + commission);
```

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

OUTPUT:

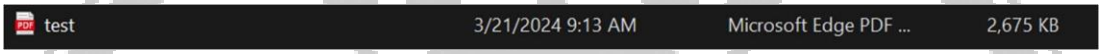
```
Enter salesman name: Joseph
Enter salesman code: 2236
Enter sales amount: 22000
Commission for salesman Joseph with code 2236 is: $2640.0
```

CYCLE 06:	Date: 22-02-2024
Networking	

1. Download the content of file from the internet

```
import java.net.*;
import java.io.*;
import java.util.Date;
class UCDemoSave
{
    public static void main(String args[]) throws Exception
    {
            int c;
            URL u = new URL
("https://mrcet.com/downloads/digital_notes/IT/JAVA%20PROGRAMMING.pdf");
            URLConnection uc = u.openConnection();
            System.out.println("Date: " + new Date(uc.getDate()));
            System.out.println("Content-Type: " + uc.getContentType());
            System.out.println("Expires: " + uc.getExpiration());
            System.out.println("Last-Modified: " + new Date(uc.getLastModified()));
            int len = uc.getContentLength();
            System.out.println("Content-Length: " + len);
            if (len > 0)
            {
                    FileOutputStream fout = new FileOutputStream("test.pdf");
                    System.out.println("=== Content ===");
                    InputStream input = uc.getInputStream();
                    int i = 0;
                    while (((c = input.read()) != -1) && i < len)
                    {
                            fout.write((char)c);
                            //System.out.print((char) c);
                            i++;
                    }
                    input.close();
                    fout.close();
            }
            else
            {
                    System.out.println("No Content Available");
            }
    }
}
```

```
F:\java> javac UCDemoSave.java
Note: UCDemoSave.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

F:\java> java UCDemoSave
Date: Thu Mar 21 09:13:35 IST 2024
Content-Type: application/pdf
Expires: 0
Last-Modified: Thu Mar 31 14:07:06 IST 2022
Content-Length: 2738664
=== Content ===
```



2. Make a public chatting program using TCP/IP

CODE:

Client

```
import java.io.*;
import java.net.*;
import java.util.*;

class ChatClient{
   public static void main(String[] args) {
      Socket cs;
      DataInputStream sin;
      DataOutputStream sout;
      Scanner s;
      try {
         cs = new Socket("localhost",1234);
         sin = new DataInputStream(cs.getInputStream());
         sout = new DataOutputStream(cs.getOutputStream());
         s = new Scanner(System.in);
         String str;
         do{
            str = sin.readUTF();
            System.out.println("Server : "+str);
            if(str.equals("quit")){
               System.out.println("Server has closed the connection!");
               break;
            }
            System.out.println("Enter text to send : ");
            str = s.nextLine();
            sout.writeUTF(str);
         }while(!str.equals("quit"));
      } catch (Exception e) {}
   }
}
```

Server

```
import java.io.*;
import java.net.*;
import java.util.*;

class ChatServer {
   public static void main(String[] args) {
      ServerSocket ss;
      Socket as;
      DataInputStream sin;
      DataOutputStream sout;
      Scanner s;
      try {
         ss = new ServerSocket(1234);
         as = ss.accept();
         sin = new DataInputStream(as.getInputStream());
```

```
            sout = new DataOutputStream(as.getOutputStream());
            s = new Scanner(System.in);
            System.out.println("Enter text to send : ");
            String str = s.nextLine();
            sout.writeUTF(str);
            while(!str.equals("quit")){
               str = sin.readUTF();
               if(str.equals("quit")){
                  ss.close();
                  System.out.println("Client has closed the connection!");
                  break;
               }
               System.out.println("Client :"+str);
               System.out.println("Enter text to send : ");
               str = s.nextLine();
               sout.writeUTF(str);
            }
         } catch (Exception e) {}
      }
   }
```

OUTPUT:

```
C:\Users\cclab25\Desktop\javaa>javac ChatServer.java

C:\Users\cclab25\Desktop\javaa>java ChatServer
Enter text to send :
hello
Client :heyy
Enter text to send :
How are you
Client :Fine. What about you
Enter text to send :
great
Client has closed the connection!
```

```
C:\Users\cclab25\Desktop\javaa>javac ChatClient.java

C:\Users\cclab25\Desktop\javaa>java ChatClient
Server : hello
Enter text to send :
heyy
Server : How are you
Enter text to send :
Fine. What about you
Server : great
Enter text to send :
quit
```

3. Make a peer to peer messaging program using UDP

CODE:

UDP Client

```
import java.net.*;
import java.io.*;
class UDPC
{
    public static void main(String args[])
    {
            DatagramSocket ds = null;
            InetAddress shost = null;
            DatagramPacket dp = null , reply = null;
            try
            {
                    ds = new DatagramSocket();
                    byte [] m = "Bye".getBytes();
                    shost = InetAddress.getByName("localhost");
                    dp =new DatagramPacket(m, 3, shost, 1234);
                    ds.send(dp);
                    byte[] buffer = new byte[1000];
                    reply = new DatagramPacket(buffer, buffer.length);
                    ds.receive(reply);
                    System.out.println("Reply: " + (new String(reply.getData())).trim());
            }
            catch (SocketException e)
            {
                    System.out.println("Socket: " + e.getMessage());
            }
            catch (IOException e)
            {
                    System.out.println("IO: " + e.getMessage());
            }
            finally
            {
                    if (ds != null)
                    ds.close();
            }
    }
}
```

UDP Server

```
import java.net.*;
import java.io.*;
class UDPS
{
    public static void main(String args[])
```

```
    {
        DatagramSocket ds = null;
        DatagramPacket dp = null, reply;
        try
        {
            ds = new DatagramSocket(1234);
            byte[] buffer = new byte[1000];
            dp = new DatagramPacket(buffer,buffer.length);
            ds.receive(dp);
            System.out.println("From Clinet: " + (new
String(dp.getData())).trim());
            System.out.println("Client PORT : "+ dp.getPort());
            reply = new DatagramPacket("From Server ok".getBytes(),"From
Server ok".length(),dp.getAddress(),dp.getPort());
            ds.send(reply);
        }
        catch (SocketException e)
        {
            System.out.println("Socket: " + e.getMessage());
        }
        catch (IOException e)
        {
            System.out.println("IO: " + e.getMessage());
        }
        finally
        {
            if (ds != null)
                ds.close();
        }
    }
}
```

OUTPUT:

```
F:\java>javac UDPS.java

F:\java>javac UDPC.java

F:\java>java UDPS
From Clinet: Bye
Client PORT : 58611
```

```
F:\java>java UDPC
Reply: From Server ok
```

CYCLE 07:	Date: 28-02-2024
Database Programming	

1. Construct the following tables:
Department (dno(Primary), dname, dloc)
Emp (eno(Primary), ename, esal ,dno(Foreign))

CODE:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;

public class CreateTablesWithJDBC {
   private static final String URL =
"jdbc:mysql://localhost:3306/java_record?characterEncoding=UTF-8";

   private static final String USER = "root";
   private static final String PASSWORD = "";

   public static void main(String[] args) {
      try {
         Class.forName("com.mysql.jdbc.Driver");
      } catch (ClassNotFoundException e) {
         System.out.println("Could not load MySQL JDBC driver.");
         e.printStackTrace();
         return;
      }

      String createDepartmentTableSQL = "CREATE TABLE Department ("
            + "dno INT PRIMARY KEY,"
            + "dname VARCHAR(255),"
            + "dloc VARCHAR(255)"
            + ")";

      String createEmpTableSQL = "CREATE TABLE Emp ("
            + "eno INT PRIMARY KEY,"
            + "ename VARCHAR(255),"
            + "esal DECIMAL(10, 2),"
            + "dno INT,"
            + "FOREIGN KEY (dno) REFERENCES Department(dno)"
            + ")";

      try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
         Statement statement = conn.createStatement()) {
```

```
            statement.executeUpdate(createDepartmentTableSQL);
            System.out.println("Department table created successfully.");
            statement.executeUpdate(createEmpTableSQL);
            System.out.println("Emp table created successfully.");

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

OUTPUT:

```
F:\java record\database>javac CreateTablesWithJDBC.java

F:\java record\database>java CreateTablesWithJDBC
Department table created successfully.
Emp table created successfully.

F:\java record\database>
```

2. Write a program for displaying information in the following order from the above tables:

eno ename esal dname dloc

101 Chetan 10,000 Civil Kochi
102 Amish 20,000 Accounts Delhi

CODE:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;

public class IDD {
   // JDBC URL, username, and password of MySQL server
   private static final String URL =
"jdbc:mysql://localhost:3306/java_record?characterEncoding=UTF-8";

   private static final String USER = "root";
   private static final String PASSWORD = "";

   public static void main(String[] args) {
      try {
         Class.forName("com.mysql.jdbc.Driver");
      } catch (ClassNotFoundException e) {
         System.out.println("Could not load MySQL JDBC driver.");
         e.printStackTrace();
         return;
      }
      insertDepartmentData();
      insertEmpData();
      displayInformation();
   }

   private static void insertDepartmentData() {
      try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
           Statement statement = conn.createStatement()) {
         String insertDept1 = "INSERT INTO Department (dno, dname, dloc) VALUES (1,
'Civil', 'Kochi')";
         String insertDept2 = "INSERT INTO Department (dno, dname, dloc) VALUES (2,
'Accounts', 'Delhi')";
         statement.executeUpdate(insertDept1);
         statement.executeUpdate(insertDept2);

         System.out.println("Department data inserted successfully.");
```

```
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    private static void insertEmpData() {
        try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
             Statement statement = conn.createStatement()) {
            String insertEmp1 = "INSERT INTO Emp (eno, ename, esal, dno) VALUES (101,
'Chetan', 10000, 1)";
            String insertEmp2 = "INSERT INTO Emp (eno, ename, esal, dno) VALUES (102,
'Amish', 20000, 2)";

            statement.executeUpdate(insertEmp1);
            statement.executeUpdate(insertEmp2);

            System.out.println("Emp data inserted successfully.");

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    private static void displayInformation() {
        try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
             Statement statement = conn.createStatement()) {

            String sql = "SELECT e.eno, e.ename, e.esal, d.dname, d.dloc " +
                    "FROM Emp e JOIN Department d ON e.dno = d.dno";
            ResultSet resultSet = statement.executeQuery(sql);
            System.out.println("eno\tename\tesal\tdname\tdloc");
            System.out.println("-------------------------------------------");
            while (resultSet.next()) {
                int eno = resultSet.getInt("eno");
                String ename = resultSet.getString("ename");
                double esal = resultSet.getDouble("esal");
                String dname = resultSet.getString("dname");
                String dloc = resultSet.getString("dloc");

                System.out.printf("%d\t%s\t%.2f\t%s\t%s\n", eno, ename, esal, dname, dloc);
            }

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

OUTPUT:

```
F:\java record\database>javac IDD.java

F:\java record\database>java IDD
Department data inserted successfully.
Emp data inserted successfully.
eno     ename   esal    dname   dloc
-------------------------------------
101     Chetan  10000.00        Civil   Kochi
102     Amish   20000.00        Accounts        Delhi

F:\java record\database>
```

3. Program to implement database connectivity using object oriented concepts.

CODE:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class OOP {
    private Connection connection;
    public OOP(String url, String username, String password) {
        try {
            connection = DriverManager.getConnection(url, username, password);
            System.out.println("Database connection established.");
        } catch (SQLException e) {
            System.out.println("Failed to establish database connection.");
            e.printStackTrace();
        }
    }
    public ResultSet executeQuery(String query) {
        try {
            PreparedStatement statement = connection.prepareStatement(query);
            return statement.executeQuery();
        } catch (SQLException e) {
            System.out.println("Error executing query: " + query);
            e.printStackTrace();
            return null;
        }
    }
    public void closeConnection() {
        try {
            if (connection != null) {
                connection.close();
                System.out.println("Database connection closed.");
            }
        } catch (SQLException e) {
            System.out.println("Error closing database connection.");
            e.printStackTrace();
        }
    }

    public void retrieveData() {
        String query = "SELECT * FROM emp";
        ResultSet resultSet = executeQuery(query);
        try {
            while (resultSet != null && resultSet.next()) {
                int eno = resultSet.getInt("eno");
```

```
            String ename = resultSet.getString("ename");
            double esal = resultSet.getDouble("esal");

            System.out.println("Employee Number: " + eno);
            System.out.println("Employee Name: " + ename);
            System.out.println("Employee Salary: " + esal);

        }
        resultSet.close();
      } catch (SQLException e) {
        System.out.println("Error retrieving data.");
        e.printStackTrace();
      }
    }

public static void main(String[] args) {
    String url = "jdbc:mysql://localhost:3306/java_record?characterEncoding=UTF-8";
    String username = "root";
    String password = "";

    try {
      Class.forName("com.mysql.jdbc.Driver");
    } catch (ClassNotFoundException e) {
      System.out.println("Could not load MySQL JDBC driver.");
      e.printStackTrace();
      return;
    }

    OOP connector = new OOP(url, username, password);
    connector.retrieveData(); // Example: retrieve data from the database
    connector.closeConnection();
}
}
```

OUTPUT :

```
F:\java record\database>javac OOP.java

F:\java record\database>java OOP
Database connection established.
Employee Number: 101
Employee Name: Chetan
Employee Salary: 10000.0
Employee Number: 102
Employee Name: Amish
Employee Salary: 20000.0
Database connection closed.

F:\java record\database>
```

4. Write a JDBC program with Parametrized queries to update a given record (Rani's salary to 15,000) in the Emp table.

CODE:

```
import java.sql.Connection;
import  java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class PARAM {
   private static final String URL =
"jdbc:mysql://localhost:3306/java_record?characterEncoding=UTF-8";
   private static final String USER = "root";
   private static final String PASSWORD = "";

   public static void main(String[] args) {
      try {
         Class.forName("com.mysql.jdbc.Driver");
      } catch (ClassNotFoundException e) {
         System.out.println("Could not load MySQL JDBC driver.");
         e.printStackTrace();
         return;
      }
      String updateQuery = "UPDATE Emp SET esal = ? WHERE ename = ?";

      try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
           PreparedStatement preparedStatement = conn.prepareStatement(updateQuery)) {

         preparedStatement.setDouble(1, 15000);
         preparedStatement.setString(2, "chetan");
         int rowsAffected = preparedStatement.executeUpdate();
         if (rowsAffected > 0) {
            System.out.println("Record updated successfully.");
         } else {
            System.out.println("No record found for Rani.");
         }

      } catch (SQLException e) {
         System.out.println("Error updating record: " + e.getMessage());
      }
   }
}
```

OUTPUT:

```
F:\java record\database>javac PARAM.java

F:\java record\database>java PARAM
Record updated successfully.

F:\java record\database>
```

5. Write a JDBC program with Parametrized queries to list the records of Emp table which has records whose names start with the alphabet "R".

CODE:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class StartR {
    private static final String URL =
"jdbc:mysql://localhost:3306/java_record?characterEncoding=UTF-8";
    private static final String USER = "root";
    private static final String PASSWORD = "";

    public static void main(String[] args) {
        try {
            Class.forName("com.mysql.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            System.out.println("Could not load MySQL JDBC driver.");
            e.printStackTrace();
            return;
        }
        String query = "SELECT * FROM Emp WHERE ename LIKE ?";

        try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
            PreparedStatement preparedStatement = conn.prepareStatement(query)) {
            preparedStatement.setString(1, "R%");
            ResultSet resultSet = preparedStatement.executeQuery();
            System.out.println("Records with names starting with 'R':");
            while (resultSet.next()) {
                int eno = resultSet.getInt("eno");
                String ename = resultSet.getString("ename");
                double esal = resultSet.getDouble("esal");
                int dno = resultSet.getInt("dno");

                System.out.println("Employee Number: " + eno);
                System.out.println("Employee Name: " + ename);
                System.out.println("Employee Salary: " + esal);
                System.out.println("Department Number: " + dno);
                System.out.println("-------------------------");
            }

        } catch (SQLException e) {
            System.out.println("Error fetching records: " + e.getMessage());
        }
    }
}
```

OUTPUT :

```
F:\java record\database>javac StartR.java

F:\java record\database>java StartR
Records with names starting with 'R':
Employee Number: 103
Employee Name: Rani
Employee Salary: 15000.0
Department Number: 3
------------------------

F:\java record\database>
```

6. Write a JDBC program with PreparedStatement to delete the records of Emp table which has records whose salary is less than 10,000.

CODE:

```
import java.sql.Connection;
import  java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class Deletion {
   private static final String URL =
"jdbc:mysql://localhost:3306/java_record?characterEncoding=UTF-8";

   private static final String USER = "root";
   private static final String PASSWORD = "";

   public static void main(String[] args) {
      try {
         Class.forName("com.mysql.jdbc.Driver");
      } catch (ClassNotFoundException e) {
         System.out.println("Could not load MySQL JDBC driver.");
         e.printStackTrace();
         return;
      }

      String query = "DELETE FROM Emp WHERE esal < ?";

      try (Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
          PreparedStatement preparedStatement = conn.prepareStatement(query)) {

         preparedStatement.setDouble(1, 10000);
         int rowsAffected = preparedStatement.executeUpdate();
         if (rowsAffected > 0) {
            System.out.println(rowsAffected + " records deleted successfully.");
         } else {
            System.out.println("No records found with salary less than 10,000.");
         }

      } catch (SQLException e) {
         System.out.println("Error deleting records: " + e.getMessage());
      }
   }
}
```

OUTPUT :

```
F:\java record\database>javac Deletion.java

F:\java record\database>java Deletion
1 records deleted successfully.

F:\java record\database>
```

7. Implement a JDBC program which uses a Stored Procedure to insert records into Department table.

CODE:

```
import   java.sql.*;
import java.util.Scanner;

public class InsertDepProcedure {
   private static final String JDBC_URL =
"jdbc:mysql://localhost:3306/java_record?characterEncoding=UTF-8";
   private static final String USERNAME = "root";
   private static final String PASSWORD = "";
   private static final String INSERT_DEPARTMENT_PROCEDURE = "{call
insert_department(?, ?, ?)}";

   public static void main(String[] args) {
      try {
         Class.forName("com.mysql.jdbc.Driver");
         try (Connection connection = DriverManager.getConnection(JDBC_URL,
USERNAME, PASSWORD);
              CallableStatement callableStatement =
connection.prepareCall(INSERT_DEPARTMENT_PROCEDURE)) {

            Scanner scanner = new Scanner(System.in);

            System.out.print("Enter Department Number: ");
            int departmentNumber = scanner.nextInt();
            scanner.nextLine();
            System.out.print("Enter Department Name: ");
            String departmentName = scanner.nextLine();
            System.out.print("Enter Department Location: ");
            String departmentLocation = scanner.nextLine();

            callableStatement.setInt(1, departmentNumber);
            callableStatement.setString(2, departmentName);
            callableStatement.setString(3, departmentLocation);
            callableStatement.executeUpdate();

            System.out.println("Department inserted successfully.");

         } catch (SQLException e) {
            e.printStackTrace();
         }
      }
         catch (ClassNotFoundException e) {
```

```
            System.out.println("Could not load MySQL JDBC driver.");
            e.printStackTrace();
        }
    }
}
```

OUTPUT :

```
C:\Users\ASUS\OneDrive\Desktop\rajagiriMCA\SEM2\javalab\cycle7>java InsertDepProcedure
Enter Department Number: 34
Enter Department Name: Youth affairs
Enter Department Location: new Delhi
Department inserted successfully.

C:\Users\ASUS\OneDrive\Desktop\rajagiriMCA\SEM2\javalab\cycle7>
```

8. Use Callable statement to implement a Stored Procedure to display the Ename and Salary of all employees.

CODE:

```
import java.sql.*;
public class DisplayEmployeeData {
   private static final String JDBC_URL =
"jdbc:mysql://localhost:3306/java_record?characterEncoding=UTF-8";
   private static final String USERNAME = "root";
   private static final String PASSWORD = "";
   private static final String GET_EMPLOYEE_DATA_PROCEDURE = "{call
get_employee_data()}";
   public static void main(String[] args) {
      try {
         Class.forName("com.mysql.jdbc.Driver");
         try (Connection connection = DriverManager.getConnection(JDBC_URL,
USERNAME, PASSWORD);
              CallableStatement callableStatement =
connection.prepareCall(GET_EMPLOYEE_DATA_PROCEDURE);
              ResultSet resultSet = callableStatement.executeQuery()) {
             System.out.println("Employee Data:");
             System.out.println("Ename \t Salary");
             while (resultSet.next()) {
                String ename = resultSet.getString("ename");
                double salary = resultSet.getDouble("esal");
                System.out.println(ename + "\t" + salary);
             }
         } catch (SQLException e) {
            e.printStackTrace();
         }
      } catch (ClassNotFoundException e) {
         System.out.println("Could not load MySQL JDBC driver.");
         e.printStackTrace();
      }
   }
}
```

OUTPUT :

```
C:\Users\ASUS\OneDrive\Desktop\rajagiriMCA\SEM2\javalab\cycle7>javac DisplayEmployeeData.java

C:\Users\ASUS\OneDrive\Desktop\rajagiriMCA\SEM2\javalab\cycle7>java DisplayEmployeeData
Employee Data:
Ename    Salary
Chetan  15000.0
Amish   20000.0
```

9. Write a JDBC program to implement Transaction Management in the Department table.

CODE:

```
import  java.sql.*;
import java.util.Scanner;

public class DepartmentTransaction {
   private static final String JDBC_URL =
"jdbc:mysql://localhost:3306/java_record?characterEncoding=UTF-8";
   private static final String USERNAME = "root";
   private static final String PASSWORD = "";

   public static void main(String[] args) {
      Connection connection = null;
      PreparedStatement preparedStatement = null;

      try {
         Class.forName("com.mysql.jdbc.Driver");
         connection = DriverManager.getConnection(JDBC_URL, USERNAME,
PASSWORD);
         connection.setAutoCommit(false);
         String sqlQuery = "INSERT INTO Department (dno, dname, dloc) VALUES (?, ?,
?)";
         preparedStatement = connection.prepareStatement(sqlQuery);
         Scanner scanner = new Scanner(System.in);
         System.out.print("Enter Department Number: ");
         int departmentNumber = scanner.nextInt();
         scanner.nextLine();
         System.out.print("Enter Department Name: ");
         String departmentName = scanner.nextLine();
         System.out.print("Enter Department Location: ");
         String departmentLocation = scanner.nextLine();
         preparedStatement.setInt(1, departmentNumber);
         preparedStatement.setString(2, departmentName);
         preparedStatement.setString(3, departmentLocation);
         preparedStatement.executeUpdate();
         connection.commit();
         System.out.println("Transaction committed successfully.");
      } catch (ClassNotFoundException e) {
         System.out.println("Could not load MySQL JDBC driver.");
         e.printStackTrace();
      } catch (SQLException e) {
         try {
            if (connection != null) {
```

```
                connection.rollback();
            }
            System.out.println("Transaction rolled back successfully.");
        } catch (SQLException ex) {
            System.out.println("Error while rolling back transaction: " + ex.getMessage());
        }
        e.printStackTrace();
    } finally {
        try {
            if (preparedStatement != null) {
                preparedStatement.close();
            }
            if (connection != null) {
                connection.close();
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
  }
}
```

OUTPUT :

```
C:\Users\ASUS\OneDrive\Desktop\rajagiriMCA\SEM2\javalab\cycle7>javac DepartmentTransaction.java

C:\Users\ASUS\OneDrive\Desktop\rajagiriMCA\SEM2\javalab\cycle7>java DepartmentTransaction
Enter Department Number: 45
Enter Department Name: electronics
Enter Department Location: trivandrum
Transaction committed successfully.
```

10. Write a JDBC program to depict the usage of SQLException Class and SQLWarning Class

CODE:

```
import java.sql.*;
public class SQLExceptionExample {
   private static final String JDBC_URL =
"jdbc:mysql://localhost:3306/java_record?characterEncoding=UTF-8";
   private static final String USERNAME = "root";
   private static final String PASSWORD = "";

   public static void main(String[] args) {
      try {
         Class.forName("com.mysql.jdbc.Driver");
         try (Connection connection = DriverManager.getConnection(JDBC_URL,
USERNAME, PASSWORD)) {
            Statement statement = connection.createStatement();
            String query = "SELECT * FROM NonExistentTable";
            ResultSet resultSet = statement.executeQuery(query);
            SQLWarning warning = statement.getWarnings();
            if (warning != null) {
               System.out.println("SQL Warning:");
               while (warning != null) {
                  System.out.println("Message: " + warning.getMessage());
                  System.out.println("SQL State: " + warning.getSQLState());
                  System.out.println("Vendor Code: " + warning.getErrorCode());
                  warning = warning.getNextWarning();
               }
            }

         } catch (SQLException e) {
            System.out.println("SQL Exception:");
            System.out.println("Message: " + e.getMessage());
            System.out.println("SQL State: " + e.getSQLState());
            System.out.println("Vendor Code: " + e.getErrorCode());
            e.printStackTrace();
         }
      } catch (ClassNotFoundException e) {
         System.out.println("Could not load MySQL JDBC driver.");
         e.printStackTrace();
      }
   }
}
```

OUTPUT :

```
C:\Users\ASUS\OneDrive\Desktop\rajagiriMCA\SEM2\javalab\cycle7>javac SQLExceptionExample.java

C:\Users\ASUS\OneDrive\Desktop\rajagiriMCA\SEM2\javalab\cycle7>java SQLExceptionExample
SQL Exception:
Message: Table 'java_record.nonexistenttable' doesn't exist
SQL State: 42S02
Vendor Code: 1146
```

CYCLE 08:	Date: 03-03-2024
Graphics Programming	

1. Write a Java program that works as a simple calculator. Arrange Buttons for digits and the + - * % operations properly. Add a text field to display the result. Handle any possible exceptions like divide by zero. Use Java Swing.

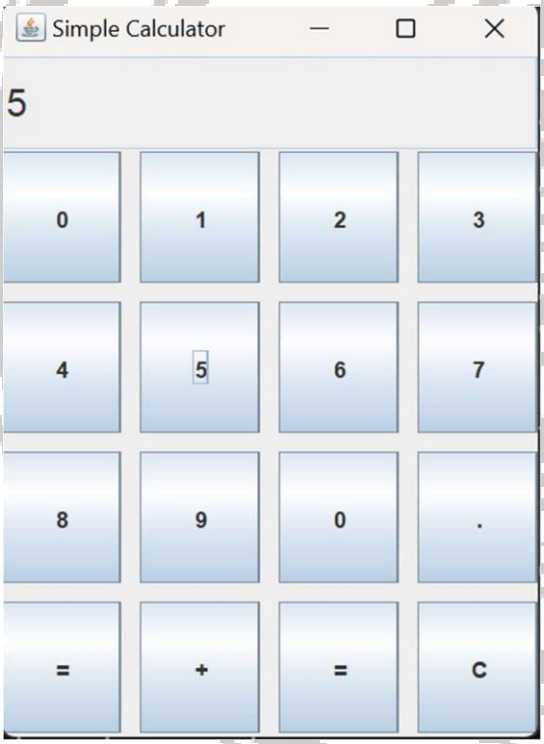
CODE:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class SimpleCalculator extends JFrame implements ActionListener {
   private JTextField textField;
   private JButton[] numberButtons;
   private JButton[] operationButtons;
   private JButton equalsButton;
   private JButton clearButton;
   private JPanel panel;
   private double firstNumber;
   private String operator;
   public SimpleCalculator() {
      setTitle("Simple Calculator");
      setSize(300, 400);
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      setLocationRelativeTo(null);
      textField = new JTextField();
      textField.setPreferredSize(new Dimension(280, 50));
      textField.setFont(new Font("Arial", Font.PLAIN, 20));
      textField.setEditable(false);
      panel = new JPanel();
      panel.setLayout(new GridLayout(4, 4, 10, 10));
      String[] buttonLabels = {
         "7", "8", "9", "/",
         "4", "5", "6", "*",
         "1", "2", "3", "-",
         "0", ".", "=", "+"
      };
      numberButtons = new JButton[10];
      for (int i = 0; i < 10; i++) {
         numberButtons[i] = new JButton(String.valueOf(i));
         numberButtons[i].addActionListener(this);
      }
      operationButtons = new JButton[4];
      for (int i = 0; i < 4; i++) {
         operationButtons[i] = new JButton(buttonLabels[i + 12]);
         operationButtons[i].addActionListener(this);
      }
```

```
        equalsButton = new JButton("=");
        equalsButton.addActionListener(this);
        clearButton = new JButton("C");
        clearButton.addActionListener(this);
        for (JButton button : numberButtons) {
            panel.add(button);
        }
        for (JButton button : operationButtons) {
            panel.add(button);
        }
        panel.add(equalsButton);
        panel.add(clearButton);
        add(textField, BorderLayout.NORTH);
        add(panel, BorderLayout.CENTER);
        setVisible(true);
    }
    public void actionPerformed(ActionEvent e) {
        String command = e.getActionCommand();
        if (command.equals("=")) {
            String secondNumberStr = textField.getText();
            double secondNumber = Double.parseDouble(secondNumberStr);
            double result = 0;
            try {
                switch (operator) {
                    case "+":
                        result = firstNumber + secondNumber;
                        break;
                    case "-":
                        result = firstNumber - secondNumber;
                        break;
                    case "*":
                        result = firstNumber * secondNumber;
                        break;
                    case "/":
                        if (secondNumber == 0) {
                            throw new ArithmeticException("Cannot divide by zero");
                        }
                        result = firstNumber / secondNumber;
                        break;
                }
                textField.setText(String.valueOf(result));
            } catch (NumberFormatException ex) {
                textField.setText("Error");
            } catch (ArithmeticException ex) {
                textField.setText("Error: " + ex.getMessage());
            }
```

```
            } else if (command.equals("C")) {
                textField.setText("");
            } else {
                textField.setText(textField.getText() + command);
            }
            if (!command.equals("=") && !command.equals("C")) {
                operator = command;
                firstNumber = Double.parseDouble(textField.getText());
            }
        }
        public static void main(String[] args) {
            new SimpleCalculator();
        }
}
```

OUTPUT:



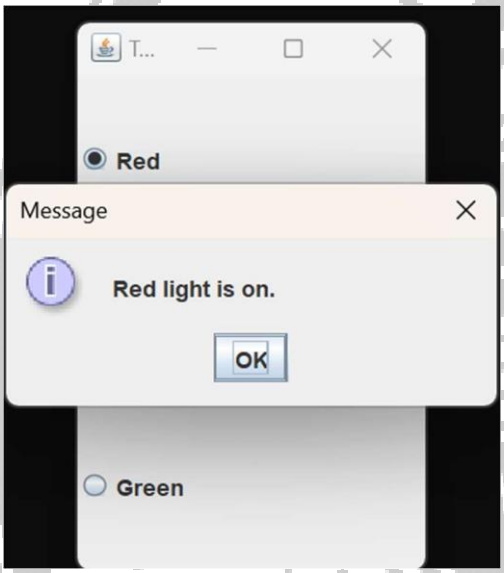
2. Write a Java program that simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green. When a radio button is selected, the light is turned on, and only one light can be on at a time. No light is on when program starts.

CODE:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class TrafficLightSimulator extends JFrame implements ActionListener {
   private JRadioButton redButton;
   private JRadioButton yellowButton;
   private JRadioButton greenButton;
   public TrafficLightSimulator() {
      setTitle("Traffic Light Simulator");
      setSize(200, 300);
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      setLocationRelativeTo(null);
      redButton = new JRadioButton("Red");
      yellowButton = new JRadioButton("Yellow");
      greenButton = new JRadioButton("Green");
      ButtonGroup group = new ButtonGroup();
      group.add(redButton);
      group.add(yellowButton);
      group.add(greenButton);
      redButton.addActionListener(this);
      yellowButton.addActionListener(this);
      greenButton.addActionListener(this);
      JPanel panel = new JPanel();
      panel.setLayout(new GridLayout(3, 1));
      panel.add(redButton);
      panel.add(yellowButton);
      panel.add(greenButton);
      add(panel);
      setVisible(true);
   }
   public void actionPerformed(ActionEvent e) {
      String command = e.getActionCommand();
      switch (command) {
         case "Red":
            showMessage("Red light is on.");
            break;
         case "Yellow":
            showMessage("Yellow light is on.");
            break;
         case "Green":
            showMessage("Green light is on.");
```

```
            break;
        }
    }
    private void showMessage(String message) {
        JOptionPane.showMessageDialog(this, message);
    }
    public static void main(String[] args) {
        new TrafficLightSimulator();
    }
}
```

OUTPUT:



CYCLE 09:	**Date: 10-03-2024**
Collection Framework	

1. Write a Java program for the following: **

1) Create a doubly linked list of elements.
2) Delete a given element from the above list.
3) Display the contents of the list after deletion.

CODE:

```
class Node {
    int data;
    Node prev;
    Node next;
    public Node(int data) {
        this.data = data;
        this.prev = null;
        this.next = null;
    }
}
class DoublyLinkedList {
    Node head;
    Node tail;
    public DoublyLinkedList() {
        this.head = null;
        this.tail = null;
    }
    public void insert(int data) {
        Node newNode = new Node(data);
        if (head == null) {
            head = newNode;
            tail = newNode;
        } else {
            tail.next = newNode;
            newNode.prev = tail;
            tail = newNode;
        }
    }
    public void delete(int data) {
        Node current = head;
        while (current != null) {
            if (current.data == data) {
                if (current.prev != null) {
                    current.prev.next = current.next;
                } else {
                    head = current.next;
```

```
                }
                if (current.next != null) {
                    current.next.prev = current.prev;
                } else {
                    tail = current.prev;
                }
                break;
            }
            current = current.next;
        }
    }
    public void display() {
        Node current = head;
        while (current != null) {
            System.out.print(current.data + " ");
            current = current.next;
        }
        System.out.println();
    }
}
public class DLL {
    public static void main(String[] args) {
        DoublyLinkedList list = new DoublyLinkedList();
        list.insert(1);
        list.insert(2);
        list.insert(3);
        list.insert(4);
        list.insert(5);
        System.out.println("Contents of the list before deletion:");
        list.display();
        int elementToDelete = 3;
        list.delete(elementToDelete);
        System.out.println("Contents of the list after deletion of " + elementToDelete + ":");
        list.display();
    }
}
```

OUTPUT:

```
F:\java>java DLL
Contents of the list before deletion:
1 2 3 4 5
Contents of the list after deletion of 3:
1 2 4 5
```

2. Write a Java program that implements Quick sort algorithm for sorting a list of names in ascending order.

CODE:

```
import java.util.Arrays;
public class QuickSortNames {
   public static void main(String[] args) {
      String[] names = {"John", "Alice", "Bob", "Eva", "Charlie", "David"};
      System.out.println("Original list of names: " + Arrays.toString(names));
      quickSort(names, 0, names.length - 1);
      System.out.println("Sorted list of names: " + Arrays.toString(names));
   }
   public static void quickSort(String[] arr, int low, int high) {
      if (low < high) {
         int pi = partition(arr, low, high);
         quickSort(arr, low, pi - 1);
         quickSort(arr, pi + 1, high);
      }
   }
   public static int partition(String[] arr, int low, int high) {
      String pivot = arr[high];
      int i = low - 1;
      for (int j = low; j < high; j++) {
         if (arr[j].compareTo(pivot) < 0) {
            i++;
            String temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
         }
      }
      String temp = arr[i + 1];
      arr[i + 1] = arr[high];
      arr[high] = temp;

      return i + 1;
   }
}
```

OUTPUT:

```
F:\java>javac QuickSortNames.java

F:\java>java QuickSortNames
Original list of names: [John, Alice, Bob, Eva, Charlie, David]
Sorted list of names: [Alice, Bob, Charlie, David, Eva, John]
```

CYCLE 10:	Date: 12-03-2024
Capstone Project	

CODE:

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

class BloodDonor {

    private static final String DB_URL =
"jdbc:mysql://localhost:3306/sarath_db?characterEncoding=utf8";
    private static final String DB_USER = "root";
    private static final String DB_PASSWORD = "Sarath@9747";

    public static void main(String[] args) {
        initializeDatabase();
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }

    private static void initializeDatabase() {
        try {
```

```
            Class.forName("com.mysql.jdbc.Driver");
            System.out.println("Connecting to database...");
            Connection connection = DriverManager.getConnection(DB_URL, DB_USER,
DB_PASSWORD);
            System.out.println("Connected to database successfully!");

            String createTableSQL = "CREATE TABLE IF NOT EXISTS donors (name
VARCHAR(255), bloodgroup VARCHAR(5), gender VARCHAR(10), weight FLOAT,
phonenumber VARCHAR(15), address VARCHAR(255), age INT)";
            try (Statement statement = connection.createStatement()) {
                statement.executeUpdate(createTableSQL);
                System.out.println("Table created successfully!");
            }
            connection.close();
        } catch (ClassNotFoundException | SQLException e) {
            e.printStackTrace();
        }
    }

    private static void createAndShowGUI() {
        JFrame frame = new JFrame("Blood Donor Management System");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 300);
        frame.setLocationRelativeTo(null);

        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(5, 1));

        JButton registerButton = new JButton("Register Donor");
        JButton searchButton = new JButton("Search Donor");
        JButton adminButton = new JButton("Admin");
```

```
JButton exitButton = new JButton("Exit");

registerButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        registerDonor();
    }
});

searchButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        searchDonor();
    }
});

adminButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        adminLogin();
    }
});

exitButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        System.exit(0);
    }
});

panel.add(registerButton);
panel.add(searchButton);
panel.add(adminButton);
panel.add(exitButton);
```

```
    frame.add(panel);
    frame.setVisible(true);
}

private static void registerDonor() {
    JFrame registerFrame = new JFrame("Register Donor");
    registerFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    registerFrame.setSize(400, 300);
    registerFrame.setLocationRelativeTo(null);

    JPanel panel = new JPanel();
    panel.setLayout(new GridLayout(8, 2));

    JLabel nameLabel = new JLabel("Name:");
    JTextField nameField = new JTextField();
    JLabel bloodGroupLabel = new JLabel("Blood Group:");
    JTextField bloodGroupField = new JTextField();
    JLabel ageLabel = new JLabel("Age:");
    JTextField ageField = new JTextField();
    JLabel genderLabel = new JLabel("Gender:");
    JTextField genderField = new JTextField();
    JLabel weightLabel = new JLabel("Weight:");
    JTextField weightField = new JTextField();
    JLabel phoneNumberLabel = new JLabel("Phone Number:");
    JTextField phoneNumberField = new JTextField();
    JLabel addressLabel = new JLabel("Address:");
    JTextField addressField = new JTextField();

    JButton registerButton = new JButton("Register");
```

```
        registerButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                String name = nameField.getText().toUpperCase();
                String bloodGroup = bloodGroupField.getText().toUpperCase();
                int age = Integer.parseInt(ageField.getText());
                String gender = genderField.getText().toUpperCase();
                float weight = Float.parseFloat(weightField.getText());
                String phoneNumber = phoneNumberField.getText();
                String address = addressField.getText().toUpperCase();

                try (Connection connection = DriverManager.getConnection(DB_URL,
DB_USER, DB_PASSWORD)) {
                    String insertSQL = "INSERT INTO donors (name, bloodgroup, gender, weight,
phonenumber, address, age) VALUES (?, ?, ?, ?, ?, ?, ?)";
                    try (PreparedStatement preparedStatement =
connection.prepareStatement(insertSQL)) {
                        preparedStatement.setString(1, name);
                        preparedStatement.setString(2, bloodGroup);
                        preparedStatement.setString(3, gender);
                        preparedStatement.setFloat(4, weight);
                        preparedStatement.setString(5, phoneNumber);
                        preparedStatement.setString(6, address);
                        preparedStatement.setInt(7, age);
                        preparedStatement.executeUpdate();
                        JOptionPane.showMessageDialog(null, "Donor registered successfully!");
                    }
                } catch (SQLException ex) {
                    ex.printStackTrace();
                    JOptionPane.showMessageDialog(null, "Error occurred while registering
donor!");
                }
            }
```

```
        });

        panel.add(nameLabel);
        panel.add(nameField);
        panel.add(bloodGroupLabel);
        panel.add(bloodGroupField);
        panel.add(ageLabel);
        panel.add(ageField);
        panel.add(genderLabel);
        panel.add(genderField);
        panel.add(weightLabel);
        panel.add(weightField);
        panel.add(phoneNumberLabel);
        panel.add(phoneNumberField);
        panel.add(addressLabel);
        panel.add(addressField);
        panel.add(new JLabel()); // Empty label for spacing
        panel.add(registerButton);

        registerFrame.add(panel);
        registerFrame.setVisible(true);
    }

    private static void searchDonor() {
        JFrame searchFrame = new JFrame("Search Donor");
        searchFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        searchFrame.setSize(600, 400);
        searchFrame.setLocationRelativeTo(null);

        JPanel panel = new JPanel(new BorderLayout());
```

```
        JPanel inputPanel = new JPanel(new FlowLayout());
        JLabel bloodGroupLabel = new JLabel("Blood Group:");
        JTextField bloodGroupField = new JTextField(10);
        JButton searchButton = new JButton("Search");
        inputPanel.add(bloodGroupLabel);
        inputPanel.add(bloodGroupField);
        inputPanel.add(searchButton);

        String[] columnNames = {"Name", "Blood Group", "Age", "Gender", "Weight", "Phone Number", "Address"};
        DefaultTableModel tableModel = new DefaultTableModel(columnNames, 0);
        JTable table = new JTable(tableModel);
        JScrollPane scrollPane = new JScrollPane(table);

        panel.add(inputPanel, BorderLayout.NORTH);
        panel.add(scrollPane, BorderLayout.CENTER);

        searchButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                String searchBloodGroup = bloodGroupField.getText().toUpperCase();
                tableModel.setRowCount(0); // Clear the table

                if (isValidBloodGroup(searchBloodGroup)) {
                    try (Connection connection = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD)) {
                        String selectSQL = "SELECT * FROM donors WHERE bloodgroup = ?";
                        try (PreparedStatement preparedStatement = connection.prepareStatement(selectSQL)) {
                            preparedStatement.setString(1, searchBloodGroup);
                            ResultSet resultSet = preparedStatement.executeQuery();
```

```
                while (resultSet.next()) {
                    String name = resultSet.getString("name");
                    String bloodGroup = resultSet.getString("bloodgroup");
                    int age = resultSet.getInt("age");
                    String gender = resultSet.getString("gender");
                    float weight = resultSet.getFloat("weight");
                    String phoneNumber = resultSet.getString("phonenumber");
                    String address = resultSet.getString("address");

                    Object[] rowData = {name, bloodGroup, age, gender, weight, phoneNumber, address};
                    tableModel.addRow(rowData);
                }

                if (tableModel.getRowCount() == 0) {
                    JOptionPane.showMessageDialog(null, "No donors found with the specified blood group.");
                }
                }
            } catch (SQLException ex) {
                ex.printStackTrace();
                JOptionPane.showMessageDialog(null, "Error occurred while searching for donors!");
            }
        } else {
            JOptionPane.showMessageDialog(null, "Invalid blood group. Please enter a valid blood group.");
        }
    }
});
```

```
        searchFrame.add(panel);
        searchFrame.setVisible(true);
    }

    private static void adminLogin() {
        JFrame adminLoginFrame = new JFrame("Admin Login");
        adminLoginFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        adminLoginFrame.setSize(300, 150);
        adminLoginFrame.setLocationRelativeTo(null);

        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(3, 2));

        JLabel usernameLabel = new JLabel("Username:");
        JTextField usernameField = new JTextField();
        JLabel passwordLabel = new JLabel("Password:");
        JPasswordField passwordField = new JPasswordField();
        JButton loginButton = new JButton("Login");

        loginButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                String username = usernameField.getText();
                String password = new String(passwordField.getPassword());
                if (username.equals("sarath") && password.equals("sarath123")) {
                    adminOperations();
                    adminLoginFrame.dispose();
                } else {
                    JOptionPane.showMessageDialog(null, "Invalid username or password. Access
denied.");
                }
            }
```

```
        });

        panel.add(usernameLabel);
        panel.add(usernameField);
        panel.add(passwordLabel);
        panel.add(passwordField);
        panel.add(new JLabel()); // Empty label for spacing
        panel.add(loginButton);

        adminLoginFrame.add(panel);
        adminLoginFrame.setVisible(true);
    }

    private static void adminOperations() {
        JFrame adminFrame = new JFrame("Admin Operations");
        adminFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        adminFrame.setSize(200, 150);
        adminFrame.setLocationRelativeTo(null);

        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(3, 1));

        JButton editButton = new JButton("Edit Donor");
        JButton deleteButton = new JButton("Delete Donor");
        JButton backButton = new JButton("Back to Main Menu");

        editButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                editDonor();
            }
```

```
    });

    deleteButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            deleteDonor();
        }
    });

    backButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            adminFrame.dispose();
        }
    });

    panel.add(editButton);
    panel.add(deleteButton);
    panel.add(backButton);

    adminFrame.add(panel);
    adminFrame.setVisible(true);
}

private static void editDonor() {
    JFrame editFrame = new JFrame("Edit Donor");
    editFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    editFrame.setSize(300, 300);
    editFrame.setLocationRelativeTo(null);

    JPanel panel = new JPanel();
    panel.setLayout(new GridLayout(4, 2));
```

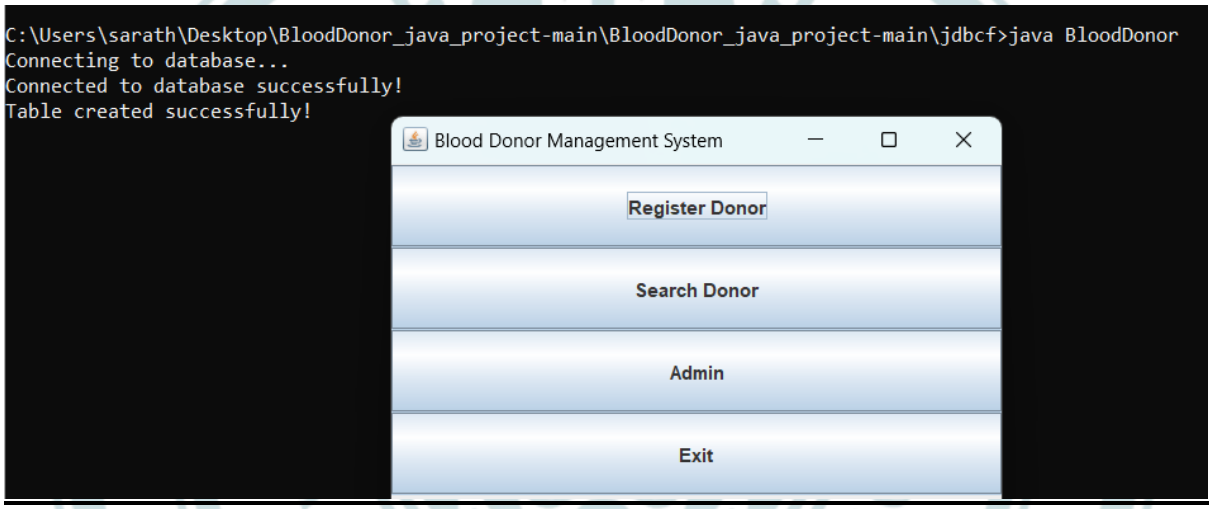
```
        JLabel nameLabel = new JLabel("Name:");
        JTextField nameField = new JTextField();
        JLabel bloodGroupLabel = new JLabel("Blood Group:");
        JTextField bloodGroupField = new JTextField();
        JButton searchButton = new JButton("Search");

        searchButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                String name = nameField.getText();
                try (Connection connection = DriverManager.getConnection(DB_URL,
DB_USER, DB_PASSWORD)) {
                    String selectSQL = "SELECT * FROM donors WHERE name = ?";
                    try (PreparedStatement preparedStatement =
connection.prepareStatement(selectSQL)) {
                        preparedStatement.setString(1, name);
                        ResultSet resultSet = preparedStatement.executeQuery();

                        if (resultSet.next()) {
                            bloodGroupField.setText(resultSet.getString("bloodgroup"));
                        } else {
                            JOptionPane.showMessageDialog(null, "Donor not found.");
                        }
                    }
                } catch (SQLException ex) {
                    ex.printStackTrace();
                    JOptionPane.showMessageDialog(null, "Error occurred while searching for
donor!");
                }
            }
        });
```

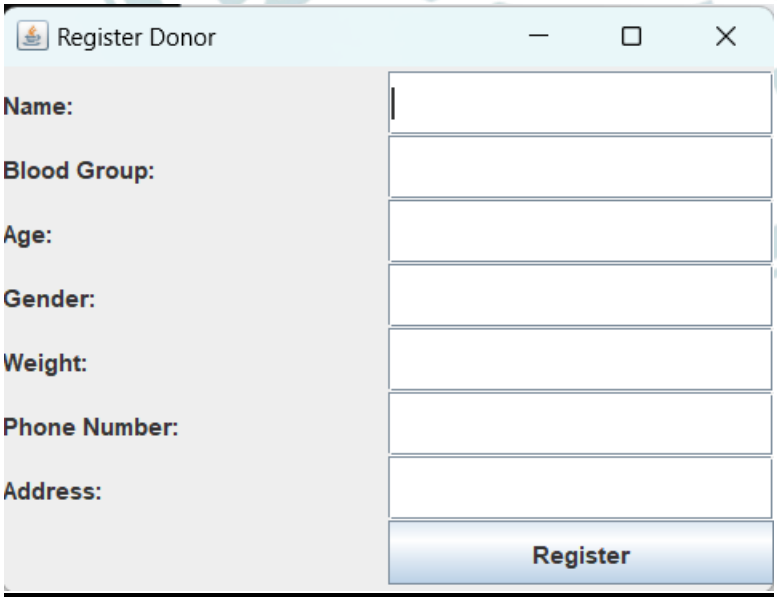
```
        panel.add(nameLabel);
        panel.add(nameField);
        panel.add(bloodGroupLabel);
        panel.add(bloodGroupField);
        panel.add(new JLabel()); // Empty label for spacing
        panel.add(searchButton);

        editFrame.add(panel);
        editFrame.setVisible(true);
    }

    private static void deleteDonor() {
        JFrame deleteFrame = new JFrame("Delete Donor");
        deleteFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        deleteFrame.setSize(300, 100);
        deleteFrame.setLocationRelativeTo(null);

        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(2, 2));

        JLabel nameLabel = new JLabel("Name:");
        JTextField nameField = new JTextField();
        JButton deleteButton = new JButton("Delete");

        deleteButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                String name = nameField.getText();
                try (Connection connection = DriverManager.getConnection(DB_URL,
DB_USER, DB_PASSWORD)) {
                    String deleteSQL = "DELETE FROM donors WHERE name = ?";
```

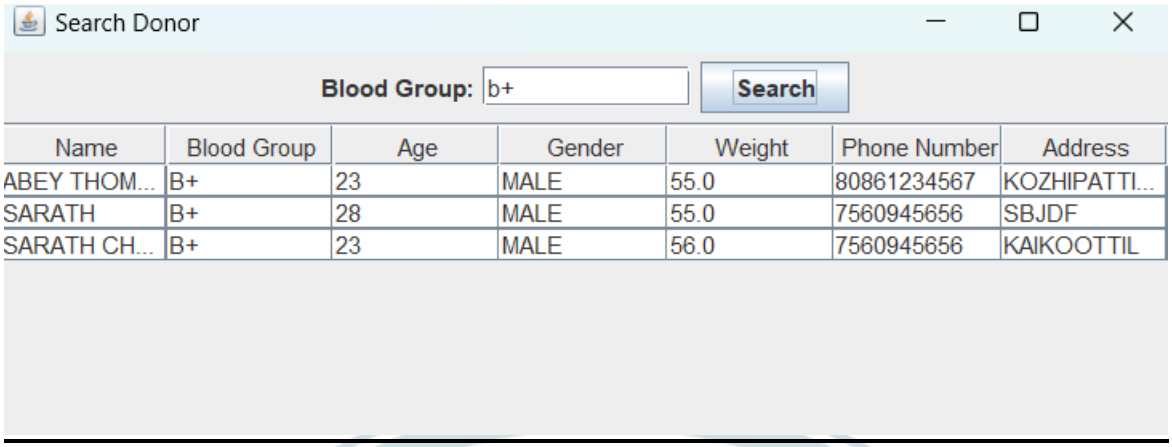
```
            try (PreparedStatement deleteStatement =
connection.prepareStatement(deleteSQL)) {
                deleteStatement.setString(1, name);
                int rowsAffected = deleteStatement.executeUpdate();
                if (rowsAffected > 0) {
                    JOptionPane.showMessageDialog(null, "Donor deleted successfully!");
                } else {
                    JOptionPane.showMessageDialog(null, "Failed to delete donor. Donor not
found.");
                }
            }
        } catch (SQLException ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(null, "Error occurred while deleting donor!");
        }
    }
});

    panel.add(nameLabel);
    panel.add(nameField);
    panel.add(new JLabel()); // Empty label for spacing
    panel.add(deleteButton);

    deleteFrame.add(panel);
    deleteFrame.setVisible(true);
}


private static boolean isValidBloodGroup(String bloodGroup) {
    String[] relevantBloodGroups = {"A+", "A-", "B+", "B-", "AB+", "AB-", "O+", "O-"};
    for (String bg : relevantBloodGroups) {
```

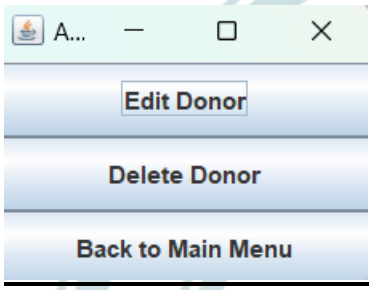
```
            if (bg.equals(bloodGroup)) {
                return true;
            }
        }
        return false;
    }
}
```

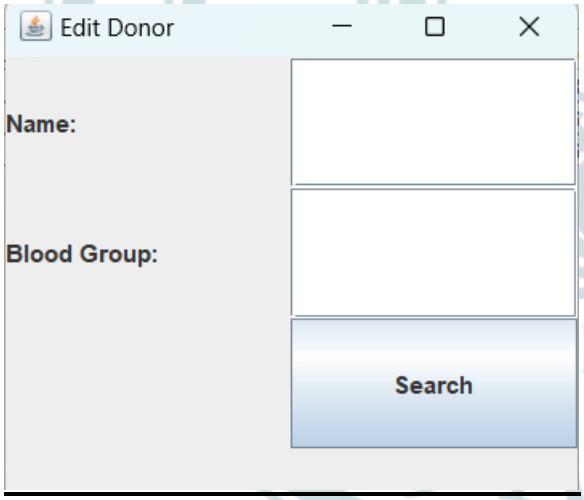
OUTPUT:






Search Donor
Blood Group: b+
Search
Name
Blood Group
Age
Gender
Weight
Phone Number
Address
ABEY THOM...
B+
23
MALE
55.0
80861234567
KOZHIPATTI...
SARATH
B+
28
MALE
55.0
7560945656
SBJDF
SARATH CH...
B+
23
MALE
56.0
7560945656
KAIKOOTTIL


A...
Edit Donor
Delete Donor
Back to Main Menu


Edit Donor
Name:
Blood Group:
Search