Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:1

Objective:

To learn about sample Java program by using class, method, variable, data type, System.out.println (), Scanner class, and format specification.

Q.1.Problem Definition:

Ajay and Vijay are two students studying Java programming. They have been given a task by their instructor to swap the values of two variables. They have to demonstrate two methods: one using a third variable, and the other without using a third variable.

- Method using a third variable:

  Ajay decides to implement the method using a third variable. He has two integer variables, x and y, with initial values x = 20 and y = 40. So, an appropriate Java program to help Ajay swap the values of x and y using a third variable, and then display the new values accordingly.

Code :

```java
public class SwapUsingThirdVariable {
    public static void main(String[] args) {
        int x = 20;
        int y = 40;
        int temp;

        // Display initial values
        System.out.println("Before swapping: x = " + x + ", y = " +
            y);

        // Swap using a third variable
        temp = x;
        x = y;
        y = temp;

        // Display swapped values
        System.out.println("After swapping: x = " + x + ", y = " + y
            );
    }
}
```

Output :

```
Before swapping: x = 20, y = 40
After swapping: x = 40, y = 20
```

- Method without using a third variable:

  Vijay prefers an approach that doesn't use an additional variable to swap his values. he also has two integer variables, a and b, with initial values a = 10 and b = 20. So, write an appropriate Java program to assist Vijay in swapping the values of a and b without using a third variable, and then display the updated values.

Code :

```java
public class SwapValues {
    public static void main(String[] args) {
        int a = 10;
        int b = 20;

        System.out.println("Before swapping:");
        System.out.println("a = " + a);
        System.out.println("b = " + b);

        // Swapping without using a third variable
        a = a + b; // a now becomes 30
        b = a - b; // b now becomes 10
        a = a - b; // a now becomes 20

        System.out.println("After swapping:");
        System.out.println("a = " + a);
        System.out.println("b = " + b);
    }
}
```

Output :

```
Before swapping:
a = 10
b = 20
After swapping:
a = 20
b = 10
```

Q.2. Write an appropriate program to find the following without the use of loop and condition.

1.) Find addition, subtraction, Multiplication, and division of given two number that is 100 and 25.

Code :

```java
public class ArithmeticOperations {
    public static void main(String[] args) {
        int num1 = 100;
        int num2 = 25;

        // Addition
        int sum = num1 + num2;

        // Subtraction
        int difference = num1 - num2;

        // Multiplication
        int product = num1 * num2;

        // Division
        int quotient = num1 / num2;

        // Display the results
        System.out.println("Addition of " + num1 + " and " + num2 + " is: " + sum);
        System.out.println("Subtraction of " + num1 + " and " + num2 + " is: " +
            difference);
        System.out.println("Multiplication of " + num1 + " and " + num2 + " is: " +
            product);
        System.out.println("Division of " + num1 + " by " + num2 + " is: " + quotient);
    }
}
```

Output :

```
Addition of 100 and 25 is: 125
Subtraction of 100 and 25 is: 75
Multiplication of 100 and 25 is: 2500
Division of 100 by 25 is: 4
```

2.) Find addition of each digit number. [Note: One number given, that is :12345
Output should be: Addition of each digit is :15

Code :

```java
public class DigitSum {
    public static void main(String[] args) {
        int number = 12345;
        int digit1 = number / 10000;
        int digit2 = (number / 1000) % 10;
        int digit3 = (number / 100) % 10;
        int digit4 = (number / 10) % 10;
        int digit5 = number % 10;

        int sum = digit1 + digit2 + digit3 + digit4 + digit5;
        System.out.println("Addition of each digit is: " + sum);
    }
}
```

Output :

```
Addition of each digit is: 15
```

3.) Find alternate digit multiplication of given one five-digit number, that is 23456. Expected output should be:720

Code :

```java
import java.util.Scanner;

class Student2 {
    public static void main(String[] args) {
        int num, rem, product = 1;
        Scanner sc = new Scanner(System.in);

        // Take user input
        System.out.print("Enter a number: ");
        num = sc.nextInt();

        // Calculate product of digits
        rem = num % 10;
        product = product * rem;
        num = num / 10;

        rem = num % 10;
        product = product * rem;
        num = num / 10;

        rem = num % 10;
        product = product * rem;
        num = num / 10;

        rem = num % 10;
        product = product * rem;
        num = num / 10;

        rem = num % 10;
        product = product * rem;
        num = num / 10;

        System.out.println("The product of the digits is: " + product);
    }
}
```

Output :

```
Enter a number: 23456
The product of the digits is: 720
```

Q.3. Charlie and James are siblings who have recently received some money as a gift from their grandparents. Charlie decides to invest his money in a fixed deposit with a simple interest rate, while James chooses to invest her money in a savings account with a compound interest rate. They want to compare the growth of their investments after a certain period.

Q.3.1: Charlie invests $2000 in a fixed deposit account with a bank that offers a simple interest rate of 4% per annum. Calculate the total amount Charlie will have after 3 years. Assume that the interest is calculated annually.

Q.3.2: James invests $1500 in a savings account with a bank that offers a compound interest rate of 5% per annum. Calculate the total amount James will have after 4 years. Assume that the interest is compounded annually.

(The formulae to calculate Simple Interest and Compound Interest are as follows:

*Simple Interest (SI) = (Principal * Rate * Time) / 100*

*Compound Interest (CI) = Principal * (1 + Rate / 100)^Time – Principal)*

Code :

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

        double charlieinvest = 2000;
        double charlieRate = 4;
        int charlieTime = 3;

        double jamesinvest = 1500;
        double jamesRate = 5;
        int jamesTime = 4;

        // Calculate and display results
        double charlieTotal = charlieinvest + (charlieinvest *
            charlieRate * charlieTime) / 100;
        double jamesTotal = jamesinvest * Math.pow((1 + jamesRate /
            100), jamesTime);

        System.out.println("Charlie's Total Amount after " +
            charlieTime + " years: $" + charlieTotal);
        System.out.println("James's Total Amount after " + jamesTime +
            " years: $" + jamesTotal);
    }
}
```

Output :

```
Charlie's Total Amount after 3 years: $2240.0
James's Total Amount after 4 years: $1823.2593
```

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:2

Objective:

To understand the concept of variable, identifier, datatypes, and program structure.

Q.1.Problem Definition:

Q.1. Assume, you are a Java Developer in a XYZ reputed company, and your team leader assigned one task, that you have to train Java Language to your junior from scratch. So, write an appropriate program to demonstrate your “Bio-data”. [Note: Perform this program on pre-initialize value as well as on user input value, it means you have to create one bio data on pre-initialize value, and then you have to create another program, where you have to take person information from the user, and then have to print that all information in bio-data format.]

Code of Pre-Initialized Bio Data :

```java
public class PreInitializedBioData {
    public static void main(String[] args) {
        // Pre-initialize values
        String name = "Vishva Patel";
        String mobileNo = "1234567890";
        String emailID = "gmail12@gmail.com";
        String gender = "Male";
        String dob = "11-01-2006";
        String nationality = "Indian";
        String religion = "Hindu";
        String fatherName = "Rakeshkumar A. Patel";
        String motherName = "Deepa R. Patel";
        String languagesKnown = "Gujarati, Hindi, English ";
        String hobbies = "Coding, Traveling, Gaming";
        String address = "11, society Name ";

        String elementarySchool = "ABC School";
        String highSchool = "XYZ High School";
        String college = "Ganpat University";
        String otherCourses = "Cybersecurity";

        String companyName1 = "IBM";
        String position1 = "Developer";
```

```
String position1 = "Developer";
String workPeriod1 = "2024-2025";

String companyName2 = "Google";
String position2 = "Software Developer";
String workPeriod2 = "2018-Present";

// Display the Bio Data
System.out.println("Bio Data:");
System.out.println("Name: " + name);
System.out.println("Mobile No: " + mobileNo);
System.out.println("Email ID: " + emailID);
System.out.println("Gender: " + gender);
System.out.println("Date of Birth: " + dob);
System.out.println("Nationality: " + nationality);
System.out.println("Religion: " + religion);
System.out.println("Father's Name: " + fatherName);
System.out.println("Mother's Name: " + motherName);
System.out.println("Languages Known: " + languagesKnown);
System.out.println("Hobbies: " + hobbies);
System.out.println("Address: " + address);
```

```
        System.out.println("\nEducational Background:");
        System.out.println("Elementary School: " + elementarySchool);
        System.out.println("High School: " + highSchool);
        System.out.println("College: " + college);
        System.out.println("Other Courses: " + otherCourses);

        System.out.println("\nWork Experience:");
        System.out.println("Company Name 1: " + companyName1);
        System.out.println("Position 1: " + position1);
        System.out.println("Work Period 1: " + workPeriod1);

        System.out.println("Company Name 2: " + companyName2);
        System.out.println("Position 2: " + position2);
        System.out.println("Work Period 2: " + workPeriod2);
    }
}
```

Output :

```
Bio Data:
Name: Vishva Patel
Mobile No: 1234567890
Email ID: gmail12@gmail.com
Gender: Male
Date of Birth: 11-01-2006
Nationality: Indian
Religion: Hindu
Father's Name: Rakeshkumar A. Patel
Mother's Name: Deepa R. Patel
Languages Known: Gujarati, Hindi, English
Hobbies: Coding, Traveling, Gaming
Address: 11, society Name

Educational Background:
Elementary School: ABC School
High School: XYZ High School
College: Ganpat University
Other Courses: Cybersecurity

Work Experience:
Company Name 1: IBM
Position 1: Developer
Work Period 1: 2024-2025
Company Name 2: Google
Position 2: Software Developer
Work Period 2: 2018-Present
```

Code of Initialized Bio Data:

```java
import java.util.Scanner;

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

        // Collect user input
        System.out.println("Enter your Bio Data details:");

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

        System.out.print("Mobile No: ");
        String mobileNo = scanner.nextLine();

        System.out.print("Email ID: ");
        String emailID = scanner.nextLine();

        System.out.print("Gender: ");
        String gender = scanner.nextLine();

        System.out.print("Date of Birth: ");
        String dob = scanner.nextLine();

        System.out.print("Nationality: ");
        String nationality = scanner.nextLine();

        System.out.print("Religion: ");
        String religion = scanner.nextLine();

        System.out.print("Father's Name: ");
        String fatherName = scanner.nextLine();
```

```
System.out.print("Mother's Name: ");
String motherName = scanner.nextLine();

System.out.print("Languages Known: ");
String languagesKnown = scanner.nextLine();

System.out.print("Hobbies: ");
String hobbies = scanner.nextLine();

System.out.print("Address: ");
String address = scanner.nextLine();

System.out.print("\nEducational Background:\n");

System.out.print("Elementary School: ");
String elementarySchool = scanner.nextLine();

System.out.print("High School: ");
String highSchool = scanner.nextLine();

System.out.print("College: ");
String college = scanner.nextLine();

System.out.print("Other Courses: ");
String otherCourses = scanner.nextLine();

System.out.print("\nWork Experience:\n");

System.out.print("Company Name 1: ");
String companyName1 = scanner.nextLine();
```

```
System.out.print("Position 1: ");
String position1 = scanner.nextLine();

System.out.print("Work Period 1: ");
String workPeriod1 = scanner.nextLine();

System.out.print("Company Name 2: ");
String companyName2 = scanner.nextLine();

System.out.print("Position 2: ");
String position2 = scanner.nextLine();

System.out.print("Work Period 2: ");
String workPeriod2 = scanner.nextLine();

// Display the Bio Data
System.out.println("\nBio Data:");
System.out.println("Name: " + name);
System.out.println("Mobile No: " + mobileNo);
System.out.println("Email ID: " + emailID);
System.out.println("Gender: " + gender);
System.out.println("Date of Birth: " + dob);
System.out.println("Nationality: " + nationality);
System.out.println("Religion: " + religion);
System.out.println("Father's Name: " + fatherName);
System.out.println("Mother's Name: " + motherName);
System.out.println("Languages Known: " + languagesKnown);
System.out.println("Hobbies: " + hobbies);
System.out.println("Address: " + address);
```

```
        System.out.println("\nEducational Background:");
        System.out.println("Elementary School: " + elementarySchool);
        System.out.println("High School: " + highSchool);
        System.out.println("College: " + college);
        System.out.println("Other Courses: " + otherCourses);

        System.out.println("\nWork Experience:");
        System.out.println("Company Name 1: " + companyName1);
        System.out.println("Position 1: " + position1);
        System.out.println("Work Period 1: " + workPeriod1);

        System.out.println("Company Name 2: " + companyName2);
        System.out.println("Position 2: " + position2);
        System.out.println("Work Period 2: " + workPeriod2);
    }
}
```

Output:

```
Enter your Bio Data details:
Name: Vishva Patel
Mobile No: 1234567890
Email ID: gmail12@gmail.com
Gender: Male
Date of Birth: 11-01-2006
Nationality: Indian
Religion: Hindu
Father's Name: Rakeshkumar Patel
Mother's Name: Deepa Patel
Languages Known: Gujarati, Hindi, English
Hobbies: Coding, Travelling, Gaming
Address: 11, xyz address

Educational Background:
Elementary School:
ABC SchoolABC School
High School: XYZ high school
College: Ganpat University
Other Courses:

Work Experience:
Company Name 1: IBM
Position 1: Software Developer
Work Period 1: 2022-2023
Company Name 2: Google
Position 2: Cyber Security Analyst
Work Period 2:
2024-Present2024-Present
```

```
Bio Data:
Name: Vishva Patel
Mobile No: 1234567890
Email ID: gmail12@gmail.com
Gender: Male
Date of Birth: 11-01-2006
Nationality: Indian
Religion: Hindu
Father's Name: Rakeshkumar Patel
Mother's Name: Deepa Patel
Languages Known: Gujarati, Hindi, English
Hobbies: Coding, Travelling, Gaming
Address: 11, xyz address

Educational Background:
Elementary School: ABC School
High School: XYZ high schoolCollege: Ganpat University
Other Courses:

Work Experience:
Company Name 1: IBM
Position 1: Software Developer
Work Period 1: 2022-2023
Company Name 2: Google
Position 2: Cyber Security Analyst
Work Period 2: 2024-Present
```

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:3

Objective:

To learn about condition (if, if-else, nested if-else, else-if ladder, switch case), class, object and constructor concept in java.

Q.1.Problem Definition:

Make a program to obtain length (L) and breadth (B) of a rectangle and check whether its area is greater, or perimeter is greater, or both are equal.

Code :

```
import java.util.Scanner;


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

        // Input the length and breadth
        int length = sc.nextInt();
        int breadth = sc.nextInt();

        // Calculate area and perimeter
        int area = length * breadth;
        int perimeter = 2 * (length + breadth);

        // Compare area and perimeter
        if (area > perimeter) {
            System.out.println("Area");
            System.out.println(area);
        } else if (perimeter > area) {
            System.out.println("Peri");
```

```
            System.out.println(perimeter);
        } else {
            System.out.println("Equal");
            System.out.println(area);
        }

        sc.close();
    }
}
```

Output :

```
7
2
Peri
18
```

Q.2. Problem definition:

Pooja would like to withdraw Rs. X from an ATM. The cash machine will only accept the transaction if X is a multiple of 5, and Pooja's account balance has enough cash to perform the withdrawal transaction (including bank charges). For each successful withdrawal the bank charges Rs. 5. Calculate Pooja's account balance after an attempted transaction.

Code :

```
import java.util.Scanner;

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

        System.out.print("Enter the withdrawal amount: ");
        int X = scanner.nextInt();
```

```
        System.out.print("Enter your initial account balance: ");
        double Y = scanner.nextDouble();

        double remainingBalance = calculateBalance(X, Y);
        System.out.printf("Your remaining balance is: %.2f\n", remainingBalance);

        scanner.close();
    }

    public static double calculateBalance(int X, double Y) {
        if (X % 5 != 0) {
            return Y; // Invalid withdrawal amount
        }

        double requiredBalance = X + 5;
        if (Y < requiredBalance) {
            return Y; // Insufficient funds
        }

        return Y - requiredBalance;
    }
}
```

Output :

Enter the withdrawal amount: 2000

Enter your initial account balance: 5000

Your remaining balance is: 2995.00

Q.3. Make a program to obtain a number N and increment its value by 1 if the number is divisible by 4,6 and 10 otherwise decrement its value by 1.

Code :

```
import java.util.Scanner;


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


        System.out.print("Enter a number: ");
        int N = scanner.nextInt();


        if (N % 4 == 0 && N % 6 == 0 && N % 10 == 0) {
            N++; // Increment by 1
        } else {
            N--; // Decrement by 1
        }


        System.out.println("New value of N: " + N);


        scanner.close();
    }
}
```

Output :

```
Enter a number: 60
New value of N: 61
```

Q.4. Compute the real roots of the equation: ax2+bx+c=0.

Code :

```
import java.util.Scanner;

public class SimpleQuadraticSolver {

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

        // Input coefficients a, b, and c
        System.out.print("Enter a: ");
        double a = scanner.nextDouble();
        System.out.print("Enter b: ");
        double b = scanner.nextDouble();
        System.out.print("Enter c: ");
        double c = scanner.nextDouble();

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

        // Determine the number of roots
        if (a == 0) {
            if (b == 0) {
                System.out.println("No solution.");
            } else {
                System.out.println("One root: " + (-c / b));
            }
        } else if (discriminant < 0) {
            System.out.println("No real roots.");
        } else if (discriminant == 0) {
```

```
            System.out.println("One root: " + (-b / (2 * a)));
        } else {
            double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
            double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);
            System.out.println("Two roots: " + root1 + " and " + root2);
        }

        scanner.close();
    }
}
```

Output :

```
Enter a: -1
Enter b: 1
Enter c:
0
Two roots: -0.0 and 1.0
```

Q.5. Determines a student's grade, for that, you have to write an appropriate program, which will read five subject marks from the user, and then have to find total and average marks of given five subjects. On the basis of the average marks, you have to find grade as per the following condition:

-if the average score =90% =>grade=O

-if the average score >=70% and <90%=> grade=A

-if the average score>=50% and <70% =>grade=B

-if the average score>=40% and <50% =>grade=C

-if the average score<40% =>grade=Fail

Code :

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

```
public class GradeCalculator {

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

        // Array to store the marks of 5 subjects
        int[] marks = new int[5];
        int total = 0;

        // Input marks for 5 subjects
        System.out.println("Enter the marks for 5 subjects:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Subject " + (i + 1) + ": ");
            marks[i] = scanner.nextInt();
            total += marks[i]; // Calculate total marks
        }

        // Calculate average marks
        double average = total / 5.0;

        // Determine the grade based on the average marks
        String grade;
        if (average >= 90) {
            grade = "O";
        } else if (average >= 70) {
            grade = "A";
        } else if (average >= 50) {
            grade = "B";
        } else if (average >= 40) {
            grade = "C";
        } else {
```

```
            grade = "Fail";
        }

        // Display the total, average, and grade
        System.out.println("Total Marks: " + total);
        System.out.println("Average Marks: " + average);
        System.out.println("Grade: " + grade);

        scanner.close();
    }
}
```

Output :

Enter the marks for 5 subjects:

Subject 1: 90

Subject 2: 92

Subject 3: 89

Subject 4: 87

Subject 5: 96

Total Marks: 454

Average Marks: 90.8

Grade: O

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:4

Objective:

To learn about switch case condition in java.

Q1. Find month name on the basis of user input month number (1 to 12).
Q2. Find weekday name on the basis of user input week-days number (1-7).
Q3. Check, whether the user number is even or odd.
Q4. Find highest and lowest number from the user input random three numbers.
Q5. Check whether the two String name is same or not.

Code :

```
import java.util.Scanner;


public class PracticalSwitchCase {


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


        do {
            System.out.println("Press <1> to find the month name by month number (1-12).");
            System.out.println("Press <2> to find the weekday name by weekday number (1-
7).");
            System.out.println("Press <3> to check whether the number is even or odd.");
            System.out.println("Press <4> to find the highest and lowest number among three
numbers.");
            System.out.println("Press <5> to check whether two strings are the same.");
            System.out.println("Press <6> to exit");
```

```
        System.out.print("Enter your choice: ");
        choice = scanner.nextInt();

        switch (choice) {
            case 1:
                findMonthName(scanner);
                break;
            case 2:
                findWeekdayName(scanner);
                break;
            case 3:
                checkEvenOdd(scanner);
                break;
            case 4:
                findHighestLowest(scanner);
                break;
            case 5:
                compareStrings(scanner);
                break;
            case 6:
                System.out.println("Exiting program.");
                break;
            default:
                System.out.println("Invalid choice! Please try again.");
        }

        System.out.println();  // Add a newline for better readability between operations

    } while (choice != 6);
```

```
        scanner.close();
    }

    // Case 1: Find month name by month number
    private static void findMonthName(Scanner scanner) {
        System.out.print("Enter a month number (1-12): ");
        int month = scanner.nextInt();

        String monthName;
        switch (month) {
            case 1:  monthName = "January"; break;
            case 2:  monthName = "February"; break;
            case 3:  monthName = "March"; break;
            case 4:  monthName = "April"; break;
            case 5:  monthName = "May"; break;
            case 6:  monthName = "June"; break;
            case 7:  monthName = "July"; break;
            case 8:  monthName = "August"; break;
            case 9:  monthName = "September"; break;
            case 10: monthName = "October"; break;
            case 11: monthName = "November"; break;
            case 12: monthName = "December"; break;
            default: monthName = "Invalid month number!"; break;
        }
        System.out.println("Month: " + monthName);
    }

    // Case 2: Find weekday name by weekday number
    private static void findWeekdayName(Scanner scanner) {
        System.out.print("Enter a weekday number (1-7): ");
        int day = scanner.nextInt();
```

```
    String dayName;
    switch (day) {
        case 1:  dayName = "Sunday"; break;
        case 2:  dayName = "Monday"; break;
        case 3:  dayName = "Tuesday"; break;
        case 4:  dayName = "Wednesday"; break;
        case 5:  dayName = "Thursday"; break;
        case 6:  dayName = "Friday"; break;
        case 7:  dayName = "Saturday"; break;
        default: dayName = "Invalid weekday number!"; break;
    }
    System.out.println("Day: " + dayName);
}

// Case 3: Check whether the number is even or odd
private static void checkEvenOdd(Scanner scanner) {
    System.out.print("Enter a number to check if it's even or odd: ");
    int number = scanner.nextInt();

    switch (number % 2) {
        case 0:
            System.out.println("The number is even.");
            break;
        case 1:
            System.out.println("The number is odd.");
            break;
        default:
            System.out.println("Error: Invalid input.");
            break;
    }
```

```
    }

    // Case 4: Find the highest and lowest number among three numbers
    private static void findHighestLowest(Scanner scanner) {
        System.out.print("Enter three numbers: ");
        int num1 = scanner.nextInt();
        int num2 = scanner.nextInt();
        int num3 = scanner.nextInt();

        int highest, lowest;

        // Finding the highest number
        if (num1 >= num2 && num1 >= num3) {
            highest = num1;
        } else if (num2 >= num1 && num2 >= num3) {
            highest = num2;
        } else {
            highest = num3;
        }

        // Finding the lowest number
        if (num1 <= num2 && num1 <= num3) {
            lowest = num1;
        } else if (num2 <= num1 && num2 <= num3) {
            lowest = num2;
        } else {
            lowest = num3;
        }

        System.out.println("Highest number: " + highest);
        System.out.println("Lowest number: " + lowest);
```

```
    }

    // Case 5: Check whether two strings are the same
    private static void compareStrings(Scanner scanner) {
        scanner.nextLine();  // Consume the newline character
        System.out.print("Enter the first string: ");
        String str1 = scanner.nextLine();
        System.out.print("Enter the second string: ");
        String str2 = scanner.nextLine();

        if (str1.equals(str2)) {
            System.out.println("The strings are the same.");
        } else {
            System.out.println("The strings are different.");
        }
    }
}
```

Output :

Press <1> to find the month name by month number (1-12).

Press <2> to find the weekday name by weekday number (1-7).

Press <3> to check whether the number is even or odd.

Press <4> to find the highest and lowest number among three numbers.

Press <5> to check whether two strings are the same.

Press <6> to exit

Enter your choice: 1

Enter a month number (1-12): 4

Month: April

Press <1> to find the month name by month number (1-12).

Press <2> to find the weekday name by weekday number (1-7).

Press <3> to check whether the number is even or odd.

Press <4> to find the highest and lowest number among three numbers.

Press <5> to check whether two strings are the same.

Press <6> to exit

Enter your choice: 2

Enter a weekday number (1-7): 4

Day: Wednesday

Press <1> to find the month name by month number (1-12).

Press <2> to find the weekday name by weekday number (1-7).

Press <3> to check whether the number is even or odd.

Press <4> to find the highest and lowest number among three numbers.

Press <5> to check whether two strings are the same.

Press <6> to exit

Enter your choice: 3

Enter a number to check if it's even or odd: 4

The number is even.

Press <1> to find the month name by month number (1-12).

Press <2> to find the weekday name by weekday number (1-7).

Press <3> to check whether the number is even or odd.

Press <4> to find the highest and lowest number among three numbers.

Press <5> to check whether two strings are the same.

Press <6> to exit

Enter your choice: 4

Enter three numbers: 4 5 6

Highest number: 6

Lowest number: 4

Press <1> to find the month name by month number (1-12).

Press <2> to find the weekday name by weekday number (1-7).

Press <3> to check whether the number is even or odd.

Press <4> to find the highest and lowest number among three numbers.

Press <5> to check whether two strings are the same.

Press <6> to exit

Enter your choice: 5

Enter the first string: hii

Enter the second string: hee

The strings are different.

Press <1> to find the month name by month number (1-12).

Press <2> to find the weekday name by weekday number (1-7).

Press <3> to check whether the number is even or odd.

Press <4> to find the highest and lowest number among three numbers.

Press <5> to check whether two strings are the same.

Press <6> to exit

Enter your choice: 6

Exiting program.

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:5

Objective:

To understand the concept of class, object, constructor and loop.

Q1. Mr. Malhotra has just started Programming, he is in first year of Engineering. Malhotra is reading about Relational Operators. Relational Operators are operators which check relationship between two values. Given two numerical values A and B you need to help Malhotra in finding the relationship between them that is, first one is greater

Code :

```java
import java.util.Scanner;

public class RelationalOperators {
    public static void main(String[] args) {
        System.out.print("Enter Number of test case:");
        Scanner scanner = new Scanner(System.in);
        int T = scanner.nextInt();  // Number of test cases

        for (int i = 0; i < T; i++) {
            int A = scanner.nextInt();
            int B = scanner.nextInt();

            if (A > B) {
                System.out.println(">");
            } else if (A < B) {
                System.out.println("<");
            } else {
                System.out.println("=");
            }
        }
        scanner.close();
    }
}
```

Output :

```
Enter Number of test case: 33
1 2
<
22 33
<
22 22
=
```

Q2. Write an appropriate program of the following.

1. Generate series of natural number like: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, ...
2. Generate table of any random number like: 5, 10, 15, 20, 25, 30, 35, 40, 45, 50,..
3. Generate series like: 6,11,16,21,26,.....
4. Generate Fibonacci series like: 0 1 1 2 3 5 8 13 21 34 .....

Code :

```java
public class SeriesGeneration {
    public static void main(String[] args) {
        generateNaturalNumbers(10);
        generateMultiplicationTable(5, 10);
        generateCustomSeries(6, 5, 10);
        generateFibonacciSeries(10);
    }

    // 1. Natural Number Series
    public static void generateNaturalNumbers(int n) {
        for (int i = 1; i <= n; i++) {
            System.out.print(i + " ");
        }
        System.out.println();
    }

    // 2. Multiplication Table
    public static void generateMultiplicationTable(int num, int length) {
        for (int i = 1; i <= length; i++) {
            System.out.print(num * i + " ");
        }
        System.out.println();
    }

    // 3. Custom Series
    public static void generateCustomSeries(int start, int step, int length) {
```

```
        for (int i = 0; i < length; i++) {
            System.out.print(start + (step * i) + " ");
        }
        System.out.println();
    }

    // 4. Fibonacci Series
    public static void generateFibonacciSeries(int length) {
        int a = 0, b = 1;
        System.out.print(a + " " + b + " ");
        for (int i = 2; i < length; i++) {
            int c = a + b;
            System.out.print(c + " ");
            a = b;
            b = c;
        }
        System.out.println();
    }
}
```

Output :

```
    1 2 3 4 5 6 7 8 9 10
    5 10 15 20 25 30 35 40 45 50
    6 11 16 21 26 31 36 41 46 51
    0 1 1 2 3 5 8 13 21 34
```

Q3. Write an appropriate program of the following.

1. Check, whether the user input one three-digit number is Armstrong number or not.
2. Check, whether the user input one three-digit number is palindrome number or not.
3. Check, whether the user input number is perfect number or not.

Code :

```
import java.util.Scanner;

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

        System.out.print("Enter a three-digit number: ");
        int num = scanner.nextInt();

        System.out.println("Is Armstrong: " + isArmstrong(num));
        System.out.println("Is Palindrome: " + isPalindrome(num));
```

```
        System.out.println("Is Perfect: " + isPerfect(num));

        scanner.close();
    }

    // Check if Armstrong number
    public static boolean isArmstrong(int num) {
        int original = num, sum = 0;
        while (num > 0) {
            int digit = num % 10;
            sum += digit * digit * digit;
            num /= 10;
        }
        return sum == original;
    }

    // Check if Palindrome
    public static boolean isPalindrome(int num) {
        int original = num, reversed = 0;
        while (num > 0) {
            int digit = num % 10;
            reversed = reversed * 10 + digit;
            num /= 10;
        }
        return reversed == original;
    }

    // Check if Perfect number
    public static boolean isPerfect(int num) {
        int sum = 0;
        for (int i = 1; i <= num / 2; i++) {
            if (num % i == 0) {
                sum += i;
            }
        }
        return sum == num;
    }
}
```

Output :

```
Enter a three-digit number: 324
Is Armstrong: false
Is Palindrome: false
Is Perfect: false
```

Q4. If Given an integer N. Write a program to obtain the sum of the first and last digits of this number.

Code :

```java
import java.util.Scanner;

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

        for (int i = 0; i < T; i++) {
            int num = scanner.nextInt();
            System.out.println(sumFirstLastDigits(num));
        }
        scanner.close();
    }

    public static int sumFirstLastDigits(int num) {
        int lastDigit = num % 10;
        while (num >= 10) {
            num /= 10;
        }
        int firstDigit = num;
        return firstDigit + lastDigit;
    }
}
```

Output :

```
3
1234
5
23454
6
34334
7
```

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:6

Objective:

To understand the concept of class, object, function, array type of object.

Q.1.Problem Definition:

Write an appropriate program, where you have to store any random 10 number from user, and then, you have to find total count of even and odd number. [Note: Perform this program by using function with and without parameter.]

Code :

```
import java.util.Scanner;

public class EvenOddCount {

    // Method without parameters
    public static void countEvenOddWithoutParams() {
        Scanner scanner = new Scanner(System.in);
        int[] numbers = new int[10];
        int evenCount = 0, oddCount = 0;

        System.out.println("Enter 10 numbers:");
        for (int i = 0; i < 10; i++) {
            numbers[i] = scanner.nextInt();
            if (numbers[i] % 2 == 0) {
                evenCount++;
            } else {
                oddCount++;
            }
        }

        System.out.println("Even count: " + evenCount);
        System.out.println("Odd count: " + oddCount);
    }

    // Method with parameters
    public static void countEvenOddWithParams(int[] numbers) {
        int evenCount = 0, oddCount = 0;

        for (int num : numbers) {
            if (num % 2 == 0) {
```

```
                evenCount++;
            } else {
                oddCount++;
            }
        }

        System.out.println("Even count: " + evenCount);
        System.out.println("Odd count: " + oddCount);
    }

    public static void main(String[] args) {
        // Using the method without parameters
        System.out.println("Counting Even and Odd numbers without
parameters:");
        countEvenOddWithoutParams();

        // Using the method with parameters
        Scanner scanner = new Scanner(System.in);
        int[] numbers = new int[10];

        System.out.println("Enter 10 numbers for the parameterized method:");
        for (int i = 0; i < 10; i++) {
            numbers[i] = scanner.nextInt();
        }

        System.out.println("Counting Even and Odd numbers with parameters:");
        countEvenOddWithParams(numbers);

        scanner.close();
    }
}
```

Output :

```
    Counting Even and Odd numbers without parameters:
    Enter 10 numbers:
    1 2 3 4 5 6 7 8 9 10
    Even count: 5
    Odd count: 5
    Enter 10 numbers for the parameterized method:
    2 3 4 5 6 7 8 9 10 11
    Counting Even and Odd numbers with parameters:
    Even count: 5
    Odd count: 5
```

Q.2. Problem definition:

Make an appropriate program of the following by using function with and without parameter.

1. Find prime number of user input number.

2. Find each digit addition of one five-digit user input number.

3. Find simple and compound interest.

4. Find reverse number of given numbers.

5. Find two number swap value.

[Note: After completing the all (1 to 5) program individually, perform this program using switch case condition also like menu driven program]

Code :

```
import java.util.Scanner;

public class MenuDrivenProgram {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int choice;
        do {
            System.out.println("Menu:");
            System.out.println("1. Find Prime Number");
            System.out.println("2. Find Digit Sum of a Five-Digit Number");
            System.out.println("3. Find Simple and Compound Interest");
            System.out.println("4. Reverse a Number");
            System.out.println("5. Swap Two Numbers");
            System.out.println("6. Exit");
            System.out.print("Enter your choice: ");
            choice = sc.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter a number to check if it's prime:
");
                    int num1 = sc.nextInt();
                    if (isPrime(num1)) {
                        System.out.println(num1 + " is a Prime number.");
                    } else {
                        System.out.println(num1 + " is not a Prime number.");
                    }
                    break;

                case 2:
                    System.out.print("Enter a five-digit number: ");
```

```
                    int num2 = sc.nextInt();
                    System.out.println("Sum of digits: " + sumOfDigits(num2));
                    break;

                case 3:
                    System.out.print("Enter principal amount: ");
                    double principal = sc.nextDouble();
                    System.out.print("Enter rate of interest: ");
                    double rate = sc.nextDouble();
                    System.out.print("Enter time period in years: ");
                    double time = sc.nextDouble();
                    System.out.println("Simple Interest: " +
simpleInterest(principal, rate, time));
                    System.out.println("Compound Interest: " +
compoundInterest(principal, rate, time));
                    break;

                case 4:
                    System.out.print("Enter a number to reverse: ");
                    int num3 = sc.nextInt();
                    System.out.println("Reversed number: " +
reverseNumber(num3));
                    break;

                case 5:
                    System.out.print("Enter first number: ");
                    int a = sc.nextInt();
                    System.out.print("Enter second number: ");
                    int b = sc.nextInt();
                    swapNumbers(a, b);
                    break;

                case 6:
                    System.out.println("Exiting...");
                    break;

                default:
                    System.out.println("Invalid choice. Please try again.");
                    break;
            }
        } while (choice != 6);

        sc.close();
    }

    // Function to check if a number is prime
    public static boolean isPrime(int num) {
        if (num <= 1) return false;
```

```
        for (int i = 2; i <= Math.sqrt(num); i++) {
            if (num % i == 0) return false;
        }
        return true;
    }

    // Function to find sum of digits of a number
    public static int sumOfDigits(int num) {
        int sum = 0;
        while (num > 0) {
            sum += num % 10;
            num /= 10;
        }
        return sum;
    }

    // Function to calculate simple interest
    public static double simpleInterest(double principal, double rate, double
time) {
        return (principal * rate * time) / 100;
    }

    // Function to calculate compound interest
    public static double compoundInterest(double principal, double rate,
double time) {
        return principal * Math.pow((1 + rate / 100), time) - principal;
    }

    // Function to reverse a number
    public static int reverseNumber(int num) {
        int reversed = 0;
        while (num != 0) {
            reversed = reversed * 10 + num % 10;
            num /= 10;
        }
        return reversed;
    }

    // Function to swap two numbers
    public static void swapNumbers(int a, int b) {
        System.out.println("Before swap: a = " + a + ", b = " + b);
        int temp = a;
        a = b;
        b = temp;
        System.out.println("After swap: a = " + a + ", b = " + b);
    }
}
```

Output :

```
 Menu:
1. Find Prime Number
2. Find Digit Sum of a Five-Digit Number
3. Find Simple and Compound Interest
4. Reverse a Number
5. Swap Two Numbers
6. Exit
Enter your choice: 1
Enter a number to check if it's prime: 2
2 is a Prime number.

 Menu:
1. Find Prime Number
2. Find Digit Sum of a Five-Digit Number
3. Find Simple and Compound Interest
4. Reverse a Number
5. Swap Two Numbers
6. Exit
Enter your choice: 2
Enter a five-digit number: 12643
Sum of digits: 16

 Menu:
1. Find Prime Number
2. Find Digit Sum of a Five-Digit Number
3. Find Simple and Compound Interest
4. Reverse a Number
5. Swap Two Numbers
6. Exit
Enter your choice: 3
Enter principal amount: 34
Enter rate of interest: 1
Enter time period in years: 2
Simple Interest: 0.68
Compound Interest: 0.6833999999999989

 Menu:
1. Find Prime Number
2. Find Digit Sum of a Five-Digit Number
3. Find Simple and Compound Interest
4. Reverse a Number
5. Swap Two Numbers
6. Exit
Enter your choice: 4
Enter a number to reverse: 345
Reversed number: 543
```

```
 Menu:
1. Find Prime Number
2. Find Digit Sum of a Five-Digit Number
3. Find Simple and Compound Interest
4. Reverse a Number
5. Swap Two Numbers
6. Exit
Enter your choice: 5
Enter first number: 22 33
Enter second number: Before swap: a = 22, b = 33
After swap: a = 33, b = 22

 Menu:
1. Find Prime Number
2. Find Digit Sum of a Five-Digit Number
3. Find Simple and Compound Interest
4. Reverse a Number
5. Swap Two Numbers
6. Exit
Enter your choice: 6
Exiting...
```

Q.3. Vivek College, which is situated at Mumbai. College authority has decided to come up with a new idea for student seating arrangement system in the exam for different-different courses. For that, college wants separate records, for those students, who one is giving a remedial exam for semester-III. For that, the college examination committee wants to take all the type of information related to students like rollno, name, class, semester, subject, and exam fee. So, whenever is required to search records by id, or by name or by class, he can search randomly. So, for the fulfilment purpose of the above said requirement make an appropriate program, where you have to store minimum five record information from the user and accordingly you have to write following function

1. Record five student information.

2. Display student record information.

3. Search information by id and name.

Code :

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

class Student {
```

```
    String id, name, className, semester, subject;
    int fee;

    Student(String id, String name, String className, String semester, String
subject, int fee) {
        this.id = id;
        this.name = name;
        this.className = className;
        this.semester = semester;
        this.subject = subject;
        this.fee = fee;
    }

    public String toString() {
        return id + " " + name + " " + className + " " + semester + " " +
subject + " " + fee;
    }
}

public class StudentManagementSystem {

    static ArrayList<Student> students = new ArrayList<>();

    public static void recordStudentInfo() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of students to record: ");
        int n = sc.nextInt();
        sc.nextLine();  // Consume newline

        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for student " + (i + 1) + ":");
            System.out.print("ID: ");
            String id = sc.nextLine();
            System.out.print("Name: ");
            String name = sc.nextLine();
            System.out.print("Class: ");
            String className = sc.nextLine();
            System.out.print("Semester: ");
            String semester = sc.nextLine();
            System.out.print("Subject: ");
            String subject = sc.nextLine();
            System.out.print("Fee: ");
            int fee = sc.nextInt();
            sc.nextLine();  // Consume newline

            students.add(new Student(id, name, className, semester, subject,
fee));
        }
```

```
    }

    public static void displayStudentInfo() {
        for (Student student : students) {
            System.out.println(student);
        }
    }

    public static void searchStudentByIdOrName() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Search by:");
        System.out.println("1. ID");
        System.out.println("2. Name");
        System.out.print("Enter choice: ");
        int choice = sc.nextInt();
        sc.nextLine();  // Consume newline

        switch (choice) {
            case 1:
                System.out.print("Enter ID to search: ");
                String id = sc.nextLine();
                for (Student student : students) {
                    if (student.id.equals(id)) {
                        System.out.println(student);
                        return;
                    }
                }
                System.out.println("Student not found.");
                break;

            case 2:
                System.out.print("Enter name to search: ");
                String name = sc.nextLine();
                for (Student student : students) {
                    if (student.name.equalsIgnoreCase(name)) {
                        System.out.println(student);
                    }
                }
                break;

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

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

```
        do {
            System.out.println("\n Menu:");
            System.out.println("1. Record Student Information");
            System.out.println("2. Display Student Information");
            System.out.println("3. Search Information by ID or Name");
            System.out.println("4. Exit");
            System.out.print("Enter your choice: ");
            choice = sc.nextInt();

            switch (choice) {
                case 1:
                    recordStudentInfo();
                    break;
                case 2:
                    displayStudentInfo();
                    break;
                case 3:
                    searchStudentByIdOrName();
                    break;
                case 4:
                    System.out.println("Exiting...");
                    break;
                default:
                    System.out.println("Invalid choice. Try again.");
            }
        } while (choice != 4);
    }
}
```

Output :

```
1. Record Student Information
2. Display Student Information
3. Search Information by ID or Name
4. Exit
Enter your choice: 1
Enter number of students to record: 2
Enter details for student 1:
ID: 21291371033
Name: vishva
Class: IT
Semester: 3
Subject: OOP
Fee: 200
Enter details for student 2:
ID: 21291371034
Name: Jayan
Class: IT
```

```
Semester: 3
Subject: OOP
Fee: 200

 Menu:
1. Record Student Information
2. Display Student Information
3. Search Information by ID or Name
4. Exit
Enter your choice: 2
21291371033 vishva IT 3 OOP 200
21291371034 Jayan IT 3 OOP 200

 Menu:
1. Record Student Information
2. Display Student Information
3. Search Information by ID or Name
4. Exit
Enter your choice: 3
Search by:
1. ID
2. Name
Enter choice: 1
Enter ID to search: 21291371033
21291371033 vishva IT 3 OOP 200

 Menu:
1. Record Student Information
2. Display Student Information
3. Search Information by ID or Name
4. Exit
Enter your choice: 4
Exiting...
```

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:7

Objective:

To learn about polymorphism (function overloading and overriding) concept in java.

Q1. Problem Definition:

Write an appropriate program of the following, by using polymorphism concept like function overloading concept.

1) Addition of two integer number.
2) Subtraction of one int number and one double number.
3) Multiplication of two double number.
4) Addition of two-character value.
5) Addition of two string value.

Code :

```
class operations {
    public int operation(int a, int b) {
        return a + b;
    }

    public double operation(int a, double b) {
        return a - b;
    }

    public double operation(double a, double b) {
        return a * b;
    }

    public int operation(char a, char b) {
        return a + b;
```

```
    }

    public String operation(String a, String b) {
        return a + b;
    }
}

public class Main {
    public static void main(String[] args) {
        operations op = new operations();

        System.out.println("Addition of two integers: " + op.operation(5, 10));
        System.out.println("Subtraction of int and double: " + op.operation(20, 5.5));
        System.out.println("Multiplication of two doubles: " + op.operation(2.5, 4.2));
        System.out.println("Addition of two characters: " + op.operation('A', 'B'));
        System.out.println("Addition of two strings: " + op.operation("Hello", " World"));
    }
}
```

Output :

Addition of two integers: 15

Subtraction of int and double: 14.5

Multiplication of two doubles: 10.5

Addition of two characters: 131

Addition of two strings: Hello World

Q2. Defination

Complete the code for the object assigned to you to satisfy the following specifications.

1) For the solving purpose of the given topic practical, you need to create minimum one class, rest as per your requirement you can take additional class also.
2) Declare minimum three function with same name and different signature as per the relevant

3) practical topic gathered information. If you want more function, you can take it as per your requirement.
4) Minimum 1 constructor method should be available in the program, rest as per your requirement.
5) You must use access specifier for data member and member function in program.
6) Wherever is required to use character data member in class, instead of that use compulsorily string data member.
7) Take minimum 5 data record information from the user and display according to the choice of user category wise. (Minimum five different options should be there for displaying information, and if you want more as per program requirement you can add more choices).
8) Use all possible filter method from stored record information:

Code :

```
import java.util.Scanner;


class librarybook {
    // Fields with access specifiers
    private int bookID;
    private char[] title = new char[30];
    private char[] author = new char[30];
    private char genre;
    private float price;

    // Constructor to initialize book details
    public librarybook(int bookID, char[] title, char[] author, char genre, float price) {
        this.bookID = bookID;
        this.title = title;
        this.author = author;
        this.genre = genre;
        this.price = price;
    }
```

```
// Overloaded methods for different operations

// Function 1: Display book details
public void displayBook() {
    System.out.println("Book ID: " + bookID);
    System.out.print("Title: ");
    System.out.println(title);
    System.out.print("Author: ");
    System.out.println(author);
    System.out.println("Genre: " + genre);
    System.out.println("Price: " + price);
}

// Function 2: Display book by genre
public void displayByGenre(char genre) {
    if (this.genre == genre) {
        displayBook();
    }
}

// Function 3: Display book by price range
public void displayByPriceRange(float minPrice, float maxPrice) {
    if (this.price >= minPrice && this.price <= maxPrice) {
        displayBook();
    }
}

// Getter methods for filters (additional, if needed)
public char getGenre() {
    return genre;
}
```

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

public class librarymanagement {
    public static void main(String[] args) {
        // Taking input for 5 books
        Scanner sc = new Scanner(System.in);
        librarybook[] books = new librarybook[5];

        for (int i = 0; i < 5; i++) {
            System.out.println("Enter details for Book " + (i + 1) + ":");
            System.out.print("Book ID: ");
            int bookID = sc.nextInt();
            sc.nextLine(); // consume the newline

            System.out.print("Title: ");
            String titleStr = sc.nextLine();
            char[] title = titleStr.toCharArray();

            System.out.print("Author: ");
            String authorStr = sc.nextLine();
            char[] author = authorStr.toCharArray();

            System.out.print("Genre (one character): ");
            char genre = sc.next().charAt(0);

            System.out.print("Price: ");
            float price = sc.nextFloat();
```

```
        // Create new book object
        books[i] = new librarybook(bookID, title, author, genre, price);
    }

    // Menu for displaying books based on user choice
    int choice;
    do {
        System.out.println("\n--- Library Book Management Menu ---");
        System.out.println("1. Display All Books");
        System.out.println("2. Display Books by Genre");
        System.out.println("3. Display Books by Price Range");
        System.out.println("4. Exit");
        System.out.print("Enter your choice: ");
        choice = sc.nextInt();

        switch (choice) {
            case 1:
                // Display all books
                for (librarybook book : books) {
                    book.displayBook();
                }
                break;

            case 2:
                // Display books by genre
                System.out.print("Enter genre to filter by (one character): ");
                char genre = sc.next().charAt(0);
                for (librarybook book : books) {
                    book.displayByGenre(genre);
                }
```

```
                break;

            case 3:
                // Display books by price range
                System.out.print("Enter minimum price: ");
                float minPrice = sc.nextFloat();
                System.out.print("Enter maximum price: ");
                float maxPrice = sc.nextFloat();
                for (librarybook book : books) {
                    book.displayByPriceRange(minPrice, maxPrice);
                }
                break;

            case 4:
                System.out.println("Exiting...");
                break;

            default:
                System.out.println("Invalid choice. Please try again.");
        }
    } while (choice != 4);

    sc.close();
  }
}
```

Output :

Enter details for Book 1:

Book ID: 1

Title: The Lord of the Rings

Author: J. R. R. Tolkien

Genre (one character): Movie

Price: 1000

Enter details for Book 2:

Book ID: 2

Title: The Godfather

Author: Mario Puzo

Genre (one character): Drama

Price: 2000

Enter details for Book 3:

Book ID: 3

Title: The Shawshank Redemption

Author: Stephen King

Genre (one character): Novel

Price: 1500

Enter details for Book 4:

Book ID: 4

Title: Harry Potter

Author: J.K. Rowling

Genre (one character): Film

Price: 1600

Enter details for Book 5:

Book ID: 5

Title: To Kill a Mockingbird

Author: Harper Lee

Genre (one character): Movie

Price: 2100

--- Library Book Management Menu ---

1. Display All Books

2. Display Books by Genre

3. Display Books by Price Range

4. Exit

Enter your choice: 1

Book ID: 1

Title: The Lord of the Rings

Author: J. R. R. Tolkien

Genre: M

Price: 1000.0

Book ID: 2

Title: The Godfather

Author: Mario Puzo

Genre: D

Price: 2000.0

Book ID: 3

Title: The Shawshank Redemption

Author: Stephen King

Genre: N

Price: 1500.0

Book ID: 4

Title: Harry Potter

Author: J.K. Rowling

Genre: F

Price: 1600.0

Book ID: 5

Title: To Kill a Mockingbird

Author: Harper Lee

Genre: M

Price: 2100.0

--- Library Book Management Menu ---

1. Display All Books

2. Display Books by Genre

3. Display Books by Price Range

4. Exit

Enter your choice: 2

Enter genre to filter by (one character): Movie

Book ID: 1

Title: The Lord of the Rings

Author: J. R. R. Tolkien

Genre: M

Price: 1000.0

Book ID: 5

Title: To Kill a Mockingbird

Author: Harper Lee

Genre: M

Price: 2100.0

--- Library Book Management Menu ---

1. Display All Books

2. Display Books by Genre

3. Display Books by Price Range

4. Exit

Enter your choice: 3

Enter minimum price: 1000

Enter maximum price: 1500

Book ID: 1

Title: The Lord of the Rings

Author: J. R. R. Tolkien

Genre: M

Price: 1000.0

Book ID: 3

Title: The Shawshank Redemption

Author: Stephen King

Genre: N

Price: 1500.0

--- Library Book Management Menu ---

1. Display All Books

2. Display Books by Genre

3. Display Books by Price Range

4. Exit

Enter your choice: 4

Exiting...

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:8

Objective:

To learn about array (1D array, 2D array, multi-dimensional array, jagged array, and object type of array).

Q1. Make a program in Java to accept any random 10 number from the user, and then find out the addition and average value of user input number by using 1D array.

Code :

```
import java.util.Scanner;


public class sumandavg {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] numbers = new int[10];
        int sum = 0;

        System.out.println("Enter 10 numbers: ");
        for (int i = 0; i < 10; i++) {
            numbers[i] = sc.nextInt();
            sum += numbers[i];
        }

        double avg = sum / 10.0;

        System.out.println("Sum: " + sum);
        System.out.println("Average: " + avg);
```

```
    }
}
```

Output :

```
Enter 10 numbers:
1 2 3 4 5 6 7 8 9 10
Sum: 55
Average: 5.5
```

Q2. Write an appropriate program to find the following without the use of loop and condition.

Code :

```
import java.util.Scanner;

public class evenodd {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] numbers = new int[10];
        int evencnt = 0;
        int oddcnt = 0;

        System.out.println("Enter 10 numbers: ");
        for (int i = 0; i < 10; i++) {
            numbers[i] = sc.nextInt();
        }

        System.out.print("Even numbers: ");
        for (int i = 0; i < 10; i++) {
            if (numbers[i] % 2 == 0) {
                System.out.print(numbers[i] + " ");
```

```
            evencnt++;
        }
    }

    System.out.print("Odd numbers: ");
    for (int i = 0; i < 10; i++) {
        if (numbers[i] % 2 != 0) {
            System.out.print(numbers[i] + " ");
            oddcnt++;
        }
    }
    System.out.println("\nTotal count of even numbers: " + evencnt);
    System.out.println("Total count of odd numbers: " + oddcnt);
  }
}
```

Output :

Enter 10 numbers:

1 2 3 4 5 6 7 8 9 10

Even numbers: 2 4 6 8 10 Odd numbers: 1 3 5 7 9

Total count of even numbers: 5

Total count of odd numbers: 5

Q3. Perform the following program with 1D array:

1. Search an element from the array list.
2. Delete an element from the array list.

Code :

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

```
public class searchdelete {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] num = new int[10];

        // Input 10 numbers
        System.out.println("Enter 10 numbers: ");
        for (int i = 0; i < 10; i++) {
            num[i] = sc.nextInt();
        }

        // Search an element
        System.out.print("Enter number to search: ");
        int searchElement = sc.nextInt();
        boolean found = false;
        for (int i = 0; i < 10; i++) {
            if (num[i] == searchElement) {
                found = true;
                System.out.println("Number found at index: " + i);
                break;
            }
        }
        if (!found) {
            System.out.println("Number not found.");
        }

        // Delete an element
        System.out.print("Enter number to delete: ");
        int deleteElement = sc.nextInt();
        found = false;
        for (int i = 0; i < 10; i++) {
```

```
            if (num[i] == deleteElement) {
                num[i] = 0; // Setting the value to 0 as a deletion
                found = true;
                System.out.println("Number deleted.");
                break;
            }
        }
        if (!found) {
            System.out.println("Number not found.");
        }
    }
}
```

Output :

Enter 10 numbers:

1 2 3 4 5 6 7 8 9 10

Enter number to search: 5

Number found at index: 4

Enter number to delete: 7

Number deleted.

Q4. Make a quiz-based character user interface program, where you have to create 5 questions, and have to give 4 option for all individual questions, and then, you should know, out of 4 option only one option will be true. So that, make an appropriate program for the fulfilment purpose of the above said requirements, and finally you have to give quiz result, on the basis of how many questions is correct or incorrect, and accordingly, you have to declare quiz result with percentage passing score.

Code :

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

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

        String[] questions = {
            "What is the capital of India?",
            "Who is known as the 'Father of the Nation' in India?",
            "Which river is considered the holiest river in India?",
            "Which Indian city is known as the 'Silicon Valley of India'?",
            "Who was the first Prime Minister of independent India?",
            "Which monument is known as the symbol of love in India?",
            "In which year did India gain independence from British rule?",
            "What is the national animal of India?",
            "Which sport is considered the most popular in India?",
            "What is the national currency of India?"
        };

        String[][] options = {
            {"1) Mumbai", "2) Kolkata", "3) New Delhi", "4) Chennai"},
            {"1) Jawaharlal Nehru", "2) Bhagat Singh", "3) Mahatma Gandhi", "4) Sardar Patel"},
            {"1) Ganga", "2) Yamuna", "3) Godavari", "4) Krishna"},
            {"1) Hyderabad", "2) Bengaluru", "3) Pune", "4) Gurugram"},
            {"1) Dr. Rajendra Prasad", "2) Subhas Chandra Bose", "3) Indira Gandhi", "4)
Jawaharlal Nehru"},
            {"1) Red Fort", "2) Qutub Minar", "3) Taj Mahal", "4) India Gate"},
            {"1) 1945", "2) 1947", "3) 1950", "4) 1965"},
            {"1) Lion", "2) Elephant", "3) Bengal Tiger", "4) Peacock"},
            {"1) Football", "2) Cricket", "3) Hockey", "4) Badminton"},
            {"1) Dollar", "2) Rupee", "3) Pound", "4) Yen"}
        };
```

```
        int[] answers = {3, 3, 1, 2, 4, 3, 2, 3, 2, 2};

        int score = 0;

        for (int i = 0; i < questions.length; i++) {
            System.out.println(questions[i]);
            for (String option : options[i]) {
                System.out.println(option);
            }
            System.out.print("Your answer: ");
            int userAnswer = sc.nextInt();
            if (userAnswer == answers[i]) {
                score++;
            }
        }

        double percentage = (score / 10.0) * 100;
        System.out.println("\nYou got " + score + " out of 10 correct.");
        System.out.println("Your score: " + percentage + "%");

        if (percentage >= 35) {
            System.out.println("Result: Pass");
        } else {
            System.out.println("Result: Fail");
        }
    }
}
```

Output :

What is the capital of India?

1) Mumbai

2) Kolkata

3) New Delhi

4) Chennai

Your answer: 4

Who is known as the 'Father of the Nation' in India?

1) Jawaharlal Nehru

2) Bhagat Singh

3) Mahatma Gandhi

4) Sardar Patel

Your answer: 3

Which river is considered the holiest river in India?

1) Ganga

2) Yamuna

3) Godavari

4) Krishna

Your answer: 2

Which Indian city is known as the 'Silicon Valley of India'?

1) Hyderabad

2) Bengaluru

3) Pune

4) Gurugram

Your answer: 3

Who was the first Prime Minister of independent India?

1) Dr. Rajendra Prasad

2) Subhas Chandra Bose

3) Indira Gandhi

4) Jawaharlal Nehru

Your answer: 2

Which monument is known as the symbol of love in India?

1) Red Fort

2) Qutub Minar

3) Taj Mahal

4) India Gate

Your answer: 3

In which year did India gain independence from British rule?

1) 1945

2) 1947

3) 1950

4) 1965

Your answer: 2

What is the national animal of India?

1) Lion

2) Elephant

3) Bengal Tiger

4) Peacock

Your answer: 3

Which sport is considered the most popular in India?

1) Football

2) Cricket

3) Hockey

4) Badminton

Your answer: 2

What is the national currency of India?

1) Dollar

2) Rupee

3) Pound

4) Yen

Your answer: 3

You got 5 out of 10 correct.

Your score: 50.0%

Result: Pass

Q5. Perform the following program:

1. Array element sorting program in ascending order.
2. Array element sorting program in descending order.

Code :

```
import java.util.Arrays;
import java.util.Collections;

public class sorting {
    public static void main(String[] args) {
        Integer[] numbers = {5, 1, 9, 3, 7, 2, 4, 6, 8, 10};

        // Ascending order
        Arrays.sort(numbers);
        System.out.println("Array in Ascending Order: " + Arrays.toString(numbers));

        // Descending order
        Arrays.sort(numbers, Collections.reverseOrder());
        System.out.println("Array in Descending Order: " + Arrays.toString(numbers));
    }
}
```

Output :

```
Array in Ascending Order: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Array in Descending Order: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
```

Q6. Perform one jagged array program as per your thoughts of knowledge.

Code :

```
public class jaggedarray {
    public static void main(String[] args) {
        // Jagged array definition
        int[][] jaggedArray = new int[3][];
        jaggedArray[0] = new int[]{1, 2};
        jaggedArray[1] = new int[]{3, 4, 5};
        jaggedArray[2] = new int[]{6, 7, 8, 9};

        // Display the elements of the jagged array
        for (int i = 0; i < jaggedArray.length; i++) {
            for (int j = 0; j < jaggedArray[i].length; j++) {
                System.out.print(jaggedArray[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

Output :

```
1 2
3 4 5
6 7 8 9
```

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:9

Objective:

To learn about class, object, constructor, control structure, loop, array, string, abstraction, encapsulation, polymorphism, switch case, interface and etc.

Q1. Make an appropriate simple project, where you have to take information (like: rollno, name, std, semester, course, subject and fee) from the user, and accordingly you have to perform the following by using switch or do-while loop condition.

1. Read minimum 5 record of given object.
2. Display all record information on screen.
3. Search record by id or name.
4. Update record by id or name.
5. Delete record by id or name.

Code :

```
import java.util.Scanner;

class Book {
    int bookID;
    String bookName;

    // Constructor
    Book(int bookID, String bookName) {
        this.bookID = bookID;
        this.bookName = bookName;
    }

    // Display Book details
    void displayBook() {
        System.out.println("Book ID: " + bookID + ", Book Name: " + bookName);
    }
}

public class LibraryManagement {
    static Scanner sc = new Scanner(System.in);
    static Book[] books = new Book[5]; // Array to store 5 book records
```

```
    public static void main(String[] args) {
        int choice;
        do {
            System.out.println("\nLibrary Management");
            System.out.println("1. Add Book");
            System.out.println("2. Display All Books");
            System.out.println("3. Search Book by ID");
            System.out.println("4. Update Book by ID");
            System.out.println("5. Delete Book by ID");
            System.out.println("6. Exit");
            System.out.print("Enter your choice: ");
            choice = sc.nextInt();

            switch (choice) {
                case 1:
                    addBooks();
                    break;
                case 2:
                    displayBooks();
                    break;
                case 3:
                    searchBook();
                    break;
                case 4:
                    updateBook();
                    break;
                case 5:
                    deleteBook();
                    break;
                case 6:
                    System.out.println("Exiting...");
                    break;
                default:
                    System.out.println("Invalid choice. Try again.");
            }
        } while (choice != 6);
    }

    // Method to add books
    public static void addBooks() {
        for (int i = 0; i < books.length; i++) {
            System.out.print("Enter Book ID: ");
            int id = sc.nextInt();
            sc.nextLine(); // Consume newline
            System.out.print("Enter Book Name: ");
            String name = sc.nextLine();
            books[i] = new Book(id, name);
```

```
        }
        System.out.println("Books added successfully.");
    }

    // Method to display all books
    public static void displayBooks() {
        for (Book book : books) {
            if (book != null) {
                book.displayBook();
            }
        }
    }

    // Method to search for a book by ID
    public static void searchBook() {
        System.out.print("Enter Book ID to search: ");
        int id = sc.nextInt();
        boolean found = false;
        for (Book book : books) {
            if (book != null && book.bookID == id) {
                book.displayBook();
                found = true;
                break;
            }
        }
        if (!found) {
            System.out.println("Book not found.");
        }
    }

    // Method to update a book's name by ID
    public static void updateBook() {
        System.out.print("Enter Book ID to update: ");
        int id = sc.nextInt();
        sc.nextLine(); // Consume newline
        boolean updated = false;

        for (Book book : books) {
            if (book != null && book.bookID == id) {
                System.out.print("Enter new Book Name: ");
                book.bookName = sc.nextLine();
                updated = true;
                System.out.println("Book updated successfully.");
                break;
            }
        }

        if (!updated) {
```

```
            System.out.println("Book not found.");
        }
    }

    // Method to delete a book by ID
    public static void deleteBook() {
        System.out.print("Enter Book ID to delete: ");
        int id = sc.nextInt();
        boolean deleted = false;
        for (int i = 0; i < books.length; i++) {
            if (books[i] != null && books[i].bookID == id) {
                books[i] = null; // Remove the book
                deleted = true;
                break;
            }
        }
        if (deleted) {
            System.out.println("Book deleted successfully.");
        } else {
            System.out.println("Book not found.");
        }
    }
}
```

Output :

Library Management

1. Add Book

2. Display All Books

3. Search Book by ID

4. Update Book by ID

5. Delete Book by ID

6. Exit

Enter your choice: 1

Enter Book ID: 1

Enter Book Name: qqq

Enter Book ID: 2

Enter Book Name: qqqq

Enter Book ID: 3

Enter Book Name: qqqqq

Enter Book ID: 4

Enter Book Name: qqqqqq

Enter Book ID: 5

Enter Book Name: qqqqqqq

Books added successfully.

Library Management

1. Add Book

2. Display All Books

3. Search Book by ID

4. Update Book by ID

5. Delete Book by ID

6. Exit

Enter your choice: 2

Book ID: 1, Book Name: qqq

Book ID: 2, Book Name: qqqq

Book ID: 3, Book Name: qqqqq

Book ID: 4, Book Name: qqqqqq

Book ID: 5, Book Name: qqqqqqq

Library Management

1. Add Book

2. Display All Books

3. Search Book by ID

4. Update Book by ID

5. Delete Book by ID

6. Exit

Enter your choice: 3

Enter Book ID to search: 2

Book ID: 2, Book Name: qqqq

Library Management

1. Add Book

2. Display All Books

3. Search Book by ID

4. Update Book by ID

5. Delete Book by ID

6. Exit

Enter your choice: 4

Enter Book ID to update: 5

Enter new Book Name: q

Book updated successfully.

Library Management

1. Add Book

2. Display All Books

3. Search Book by ID

4. Update Book by ID

5. Delete Book by ID

6. Exit

Enter your choice: 2

Book ID: 1, Book Name: qqq

Book ID: 2, Book Name: qqqq

Book ID: 3, Book Name: qqqqq

Book ID: 4, Book Name: qqqqqq

Book ID: 5, Book Name: q

Library Management

1. Add Book

2. Display All Books

3. Search Book by ID

4. Update Book by ID

5. Delete Book by ID

6. Exit

Enter your choice: 5

Enter Book ID to delete: 1

Book deleted successfully.

Library Management

1. Add Book

2. Display All Books

3. Search Book by ID

4. Update Book by ID

5. Delete Book by ID

6. Exit

Enter your choice: 2

Book ID: 2, Book Name: qqqq

Book ID: 3, Book Name: qqqqq

Book ID: 4, Book Name: qqqqqq

Book ID: 5, Book Name: q

Library Management

1. Add Book

2. Display All Books

3. Search Book by ID

4. Update Book by ID

5. Delete Book by ID

6. Exit

Enter your choice: 6

Exiting...

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number: 10

Objective:

To learn about inheritance concepts in java.

Q1. Make an appropriate simple “Account management” project based practical by using inheritance concepts like (Single L. Inheritance, Multi L. Inheritance , Hierarchical L. Inheritance and Hybrid L. Inheritance)

In project, you have to include the following modules:

1. Create customer accounts in the bank.
2. Deposit amount in Account.
3. Withdraw amount from Account.
4. Transfer amount from one account to another account.
5. Display balance by Account ID or Name
6. Modify customer account information.
7. Delete Account
8. Display monthly transaction Report.

- **Note:1**: You have to create minimum five customer account in bank and then have to perform all others module accordingly.
- **Note:2**: After using inheritance concept, you can keep all your modules in switch case condition or in a do-while loop condition for a menu driven or choice based program.

Code :

```
import java.util.ArrayList;

import java.util.Scanner;


// Base class

class Account {

    protected int accountId;
```

```
    protected String accountHolderName;
    protected double balance;

    public Account(int accountId, String accountHolderName, double balance) {
        this.accountId = accountId;
        this.accountHolderName = accountHolderName;
        this.balance = balance;
    }

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

    public void withdraw(double amount) {
        if (amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawal Successful! New Balance: " + balance);
        } else {
            System.out.println("Insufficient Balance!");
        }
    }

    public double getBalance() {
        return balance;
    }

    public String getAccountHolderName() {
        return accountHolderName;
    }
```

```
    public int getAccountId() {
        return accountId;
    }
}

// SavingsAccount and CurrentAccount (Hierarchical Inheritance)
class SavingsAccount extends Account {
    public SavingsAccount(int accountId, String accountHolderName, double balance) {
        super(accountId, accountHolderName, balance);
    }
}

class CurrentAccount extends Account {
    public CurrentAccount(int accountId, String accountHolderName, double balance) {
        super(accountId, accountHolderName, balance);
    }
}

// Transaction class (Multilevel Inheritance)
class Transaction extends SavingsAccount {
    public Transaction(int accountId, String accountHolderName, double balance) {
        super(accountId, accountHolderName, balance);
    }

    public void transfer(Account recipient, double amount) {
        if (amount <= balance) {
            this.withdraw(amount);
            recipient.deposit(amount);
            System.out.println("Transfer Successful!");
        } else {
            System.out.println("Insufficient Balance for Transfer!");
```

```
        }
    }
}

// AccountManager (Hybrid Inheritance)
class AccountManager {
    private ArrayList<Account> accounts = new ArrayList<>();

    public void createAccount(String type, int accountId, String name, double initialBalance) {
        Account account = type.equals("Savings") ? new SavingsAccount(accountId, name,
initialBalance)
                                        : new CurrentAccount(accountId, name, initialBalance);
        accounts.add(account);
        System.out.println(type + " Account Created Successfully!");
    }

    public Account findAccount(int accountId) {
        for (Account account : accounts) {
            if (account.getAccountId() == accountId) {
                return account;
            }
        }
        return null;
    }

    public void deleteAccount(int accountId) {
        Account account = findAccount(accountId);
        if (account != null) {
            accounts.remove(account);
            System.out.println("Account Deleted Successfully!");
        } else {
```

```
            System.out.println("Account Not Found!");
        }
    }

    public void displayAccount(int accountId) {
        Account account = findAccount(accountId);
        if (account != null) {
            System.out.println("Account ID: " + account.getAccountId() +
                               ", Name: " + account.getAccountHolderName() +
                               ", Balance: " + account.getBalance());
        } else {
            System.out.println("Account Not Found!");
        }
    }

    public void modifyAccount(int accountId, String newName) {
        Account account = findAccount(accountId);
        if (account != null) {
            account.accountHolderName = newName;
            System.out.println("Account Modified Successfully!");
        } else {
            System.out.println("Account Not Found!");
        }
    }

    public void displayMonthlyReport() {
        System.out.println("Monthly Transaction Report:");
        for (Account account : accounts) {
            System.out.println("Account ID: " + account.getAccountId() +
                               ", Name: " + account.getAccountHolderName() +
                               ", Balance: " + account.getBalance());
```

```
        }
    }
}

// Main Class for Menu-driven Approach
public class BankManagementSystem {
    public static void main(String[] args) {
        AccountManager accountManager = new AccountManager();
        Scanner scanner = new Scanner(System.in);
        int choice;

        do {
            System.out.println("\n1. Create Account\n2. Deposit\n3. Withdraw\n4. Transfer\n5. Display Account\n6. Modify Account\n7. Delete Account\n8. Monthly Report\n9. Exit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter Account Type (Savings/Current): ");
                    String type = scanner.next();
                    System.out.print("Enter Account ID: ");
                    int id = scanner.nextInt();
                    System.out.print("Enter Account Holder Name: ");
                    String name = scanner.next();
                    System.out.print("Enter Initial Balance: ");
                    double balance = scanner.nextDouble();
                    accountManager.createAccount(type, id, name, balance);
                    break;
                case 2:
                    System.out.print("Enter Account ID: ");
```

```
            id = scanner.nextInt();
            System.out.print("Enter Amount to Deposit: ");
            double depositAmount = scanner.nextDouble();
            Account account = accountManager.findAccount(id);
            if (account != null) {
                account.deposit(depositAmount);
            } else {
                System.out.println("Account Not Found!");
            }
            break;
        case 3:
            System.out.print("Enter Account ID: ");
            id = scanner.nextInt();
            System.out.print("Enter Amount to Withdraw: ");
            double withdrawAmount = scanner.nextDouble();
            account = accountManager.findAccount(id);
            if (account != null) {
                account.withdraw(withdrawAmount);
            } else {
                System.out.println("Account Not Found!");
            }
            break;
        case 4:
            System.out.print("Enter Sender Account ID: ");
            int senderId = scanner.nextInt();
            System.out.print("Enter Recipient Account ID: ");
            int recipientId = scanner.nextInt();
            System.out.print("Enter Transfer Amount: ");
            double transferAmount = scanner.nextDouble();
            Account sender = accountManager.findAccount(senderId);
            Account recipient = accountManager.findAccount(recipientId);
```

```
            if (sender != null && recipient != null) {
                ((Transaction) sender).transfer(recipient, transferAmount);
            } else {
                System.out.println("Account Not Found!");
            }
            break;
        case 5:
            System.out.print("Enter Account ID: ");
            id = scanner.nextInt();
            accountManager.displayAccount(id);
            break;
        case 6:
            System.out.print("Enter Account ID: ");
            id = scanner.nextInt();
            System.out.print("Enter New Account Holder Name: ");
            name = scanner.next();
            accountManager.modifyAccount(id, name);
            break;
        case 7:
            System.out.print("Enter Account ID to Delete: ");
            id = scanner.nextInt();
            accountManager.deleteAccount(id);
            break;
        case 8:
            accountManager.displayMonthlyReport();
            break;
        case 9:
            System.out.println("Exiting System...");
            break;
        default:
            System.out.println("Invalid Choice! Please try again.");
```

```
            }
        } while (choice != 9);

        scanner.close();
    }
}
```

Output :

1. Create Account
2. Deposit
3. Withdraw
4. Transfer
5. Display Account
6. Modify Account
7. Delete Account
8. Monthly Report
9. Exit

Enter your choice: 1

Enter Account Type (Savings/Current): Savings

Enter Account ID: 1

Enter Account Holder Name: vishva

Enter Initial Balance: 50000

Savings Account Created Successfully!

1. Create Account
2. Deposit
3. Withdraw
4. Transfer
5. Display Account
6. Modify Account
7. Delete Account
8. Monthly Report

9. Exit

Enter your choice: 1

Enter Account Type (Savings/Current): Current

Enter Account ID: 2

Enter Account Holder Name: Aryan

Enter Initial Balance: 100000

Current Account Created Successfully!

1. Create Account

2. Deposit

3. Withdraw

4. Transfer

5. Display Account

6. Modify Account

7. Delete Account

8. Monthly Report

9. Exit

Enter your choice: 1

Enter Account Type (Savings/Current): savings

Enter Account ID: 3

Enter Account Holder Name: jeal

Enter Initial Balance: 50000

savings Account Created Successfully!

1. Create Account

2. Deposit

3. Withdraw

4. Transfer

5. Display Account

6. Modify Account

7. Delete Account

8. Monthly Report

9. Exit

Enter your choice: 1

Enter Account Type (Savings/Current): savings

Enter Account ID: 4

Enter Account Holder Name: rishi

Enter Initial Balance: 50000

savings Account Created Successfully!

1. Create Account

2. Deposit

3. Withdraw

4. Transfer

5. Display Account

6. Modify Account

7. Delete Account

8. Monthly Report

9. Exit

Enter your choice: 1

Enter Account Type (Savings/Current): savings

Enter Account ID: 5

Enter Account Holder Name: Nisarg

Enter Initial Balance: 50000

savings Account Created Successfully!

1. Create Account

2. Deposit

3. Withdraw

4. Transfer

5. Display Account

6. Modify Account

7. Delete Account

8. Monthly Report

9. Exit

Enter your choice: 2

Enter Account ID: 1

Enter Amount to Deposit: 50000

Deposit Successful! New Balance: 100000.0

1. Create Account

2. Deposit

3. Withdraw

4. Transfer

5. Display Account

6. Modify Account

7. Delete Account

8. Monthly Report

9. Exit

Enter your choice: 3

Enter Account ID: 2

Enter Amount to Withdraw: 50000

Withdrawal Successful! New Balance: 50000.0

1. Create Account

2. Deposit

3. Withdraw

4. Transfer

5. Display Account

6. Modify Account

7. Delete Account

8. Monthly Report

9. Exit

Enter your choice: 4

Enter Sender Account ID: 1

Enter Recipient Account ID: 3

Enter Transfer Amount: 25000

Withdrawal Successful! New Balance: 75000.0

Deposit Successful! New Balance: 75000.0

Transfer Successful!

1. Create Account
2. Deposit
3. Withdraw
4. Transfer
5. Display Account
6. Modify Account
7. Delete Account
8. Monthly Report
9. Exit

Enter your choice: 5

Enter Account ID: 1

Account ID: 1, Name: vishva, Balance: 75000.0

1. Create Account
2. Deposit
3. Withdraw
4. Transfer
5. Display Account
6. Modify Account
7. Delete Account
8. Monthly Report
9. Exit

Enter your choice: 6

Enter Account ID: 1

Enter New Account Holder Name: Vishva_Patel

Account Modified Successfully!

1. Create Account

2. Deposit

3. Withdraw

4. Transfer

5. Display Account

6. Modify Account

7. Delete Account

8. Monthly Report

9. Exit

Enter your choice: 7

Enter Account ID to Delete: 5

Account Deleted Successfully!

1. Create Account

2. Deposit

3. Withdraw

4. Transfer

5. Display Account

6. Modify Account

7. Delete Account

8. Monthly Report

9. Exit

Enter your choice: 8

Monthly Transaction Report:

Account ID: 1, Name: Vishva_Patel, Balance: 75000.0

Account ID: 2, Name: Aryan, Balance: 50000.0

Account ID: 3, Name: jeal, Balance: 75000.0

Account ID: 4, Name: rishi, Balance: 50000.0

1. Create Account
2. Deposit
3. Withdraw
4. Transfer
5. Display Account
6. Modify Account
7. Delete Account
8. Monthly Report
9. Exit

Enter your choice: 9

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number: 11

Objective:

To learn about exception handling concepts in Java.

Q1. Explain exception, and exception handling concept in java.

ANS :

Exception: An exception is an event that disrupts the normal flow of a program during runtime. It is an object representing an error condition.

Exception Handling: It is a mechanism to handle runtime errors, ensuring the normal flow of the program.

Q2. What is the difference between exception and error in java? Explain with an example.

ANS :

Exception	Error
Recoverable at runtime.	Unrecoverable (e.g., OutOfMemoryError).
Handled using `try-catch`.	Rarely handled.
Examples: NullPointerException, IOException.	Examples: StackOverflowError, VirtualMachineError.

Q3. Explain the concept of try and catch block in java exception handling with an appropriate program example.

Code :

```
public class TryCatchExample {
    public static void main(String[] args) {
        try {
            int data = 50 / 0;
        } catch (ArithmeticException e) {
```

```
            System.out.println("Caught Exception: " + e);
        }
    }
}
```

Q4. Explain throw and throws concept in java exception handling with an appropriate program example.

ANS :

Throw: Used to explicitly throw an exception.

Throws: Used in method declaration to indicate exceptions the method can throw.

E.g

```
class ThrowThrowsExample {
    void checkAge(int age) throws IllegalArgumentException {
        if (age < 18) {
            throw new IllegalArgumentException("Age is not valid!");
        }
    }
    public static void main(String[] args) {
        try {
            new ThrowThrowsExample().checkAge(15);
        } catch (IllegalArgumentException e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```

Q5. Explain the concept of multiple catch block in java exception handling with an appropriate program example.

Code :

```
public class MultipleCatchExample {
    public static void main(String[] args) {
        try {
            int[] arr = new int[5];
            arr[5] = 10 / 0;
        } catch (ArithmeticException e) {
            System.out.println("Arithmetic Exception");
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Array Index Out of Bounds");
        }
    }
}
```

Q6. Explain the concept of finally block in java exception handling with an example.

Code :

```
public class FinallyExample {
    public static void main(String[] args) {
        try {
            int data = 10 / 0;
        } catch (ArithmeticException e) {
            System.out.println("Exception: " + e.getMessage());
        } finally {
            System.out.println("Finally block executed");
        }
    }
}
```

Error :

OutOfMemoryError

Q7. Explain hierarchy of the exception class with an appropriate diagram example.

Code :

Output :

Q8. Write an appropriate program of the following checked exception

1. IOException.

Code :
```
import java.io.*;

public class IOExceptionExample {
    public static void main(String[] args) {
        try {
            FileReader fr = new FileReader("nonexistent.txt");
        } catch (IOException e) {
            System.out.println("IOException occurred: " + e.getMessage());
        }
    }
}
```

2. FileNotFoundException.

Code :
```
import java.io.*;
public class FileNotFoundExceptionExample {
    public static void main(String[] args) {
        try {
            FileReader fr = new FileReader("missingfile.txt");
        } catch (FileNotFoundException e) {
            System.out.println("FileNotFoundException occurred: " + e.getMessage());
```

```
        }
    }
}
```

3. ClassNotFoundException.

Code :

```
public class ClassNotFoundExceptionExample {
    public static void main(String[] args) {
        try {
            Class.forName("NonExistentClass");
        } catch (ClassNotFoundException e) {
            System.out.println("ClassNotFoundException occurred: " + e.getMessage());
        }
    }
}
```

4. SQLException.

Code :

```
import java.sql.*;
public class SQLExceptionExample {
    public static void main(String[] args) {
        try {
            Connection conn = DriverManager.getConnection("invalid-url", "user", "pass");
        } catch (SQLException e) {
            System.out.println("SQLException occurred: " + e.getMessage());
        }
    }
}
```

5. InterruptedException

Code :
```
public class InterruptedExceptionExample {
    public static void main(String[] args) {
        Thread t = new Thread(() -> {});
        t.start();
        try {
            t.join();
        } catch (InterruptedException e) {
            System.out.println("InterruptedException occurred: " + e.getMessage());
        }
    }
}
```

6. Instantiation Exception

Code :
```
public class InstantiationExceptionExample {
    public static void main(String[] args) {
        try {
            Class<?> clazz = Class.forName("java.util.ArrayList");
            Object obj = clazz.newInstance();
        } catch (InstantiationException | IllegalAccessException | ClassNotFoundException e) {
            System.out.println("Exception occurred: " + e.getMessage());
        }
    }
}
```

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:12

Objective: To learn about exception handling concepts in Java.

Problem Definition:

Q.1. Write an appropriate program of the following Unchecked (runtime) exception

1. Arithmetic Exception.

2. ArrayIndexOutOfBoundException.

3. ClassCastException.

4. IllegalParameterException.

E.g

```
import java.util.Scanner;

public class UncheckedExceptionExamples {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.println("Select an exception to demonstrate:");
      System.out.println("1. ArithmeticException");
      System.out.println("2. ArrayIndexOutOfBoundsException");
      System.out.println("3. ClassCastException");
      System.out.println("4. IllegalArgumentException");
      int choice = scanner.nextInt();

      switch (choice) {
         case 1:
            try {
               int result = 10 / 0; // Division by zero
            } catch (ArithmeticException e) {
```

```
            System.out.println("ArithmeticException occurred: " +
e.getMessage());
        }
        break;

    case 2:
        try {
            int[] arr = {1, 2, 3};
            System.out.println(arr[5]); // Invalid index
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("ArrayIndexOutOfBoundsException
occurred: " + e.getMessage());
        }
        break;

    case 3:
        try {
            Object obj = new Integer(100);
            String str = (String) obj; // Invalid casting
        } catch (ClassCastException e) {
            System.out.println("ClassCastException occurred: " +
e.getMessage());
        }
        break;

    case 4:
        try {
            Thread t = new Thread();
            t.setPriority(20); // Invalid priority
        } catch (IllegalArgumentException e) {
```

```
                System.out.println("IllegalArgumentException occurred: "
+ e.getMessage());
            }
            break;

        default:
            System.out.println("Invalid choice!");
            break;
      }
      scanner.close();
   }
}
```

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:13

Objective: To learn about threading concepts in Java.

Problem Definition:

Q.1. What is the concept of threading in Java? Explain with the help of a sample program example.

Ans:- Threading in Java allows the execution of multiple threads (small units of a process) concurrently. It enhances the efficiency of CPU utilization and allows multitasking.

E.g

```
class MyThread extends Thread {
   public void run() {
      System.out.println("Thread is running...");
   }
}

public class ThreadingExample {
   public static void main(String[] args) {
      MyThread t = new MyThread();
      t.start(); // Starts the thread
   }
}
```

Q.2. Explain single and multithreading in java with real life examples.

Ans:- Single-threading: Only one task executes at a time. Example: Reading a file line by line.
Multithreading: Multiple tasks execute simultaneously. Example: Listening to music while downloading a file.
Real-life Example:

- Single-threading: A person cooking one dish at a time.

- Multithreading: A person cooking multiple dishes simultaneously.

Q.3. Write an appropriate program by using the following method:

1. **run() method.**

Ans:-
```
class MyThread extends Thread {
   public void run() {
      System.out.println("Running thread...");
   }
}

public class RunMethodExample {
   public static void main(String[] args) {
      MyThread t = new MyThread();
      t.start();
   }
}
```

2. **start() method.**

Ans:-
```
class MyThread extends Thread {
   public void run() {
      System.out.println("Thread started!");
   }
}

public class StartMethodExample {
   public static void main(String[] args) {
      MyThread t = new MyThread();
      t.start();
   }
}
```

3. sleep() method.

Ans:-
```
class SleepExample extends Thread {
   public void run() {
      for (int i = 1; i <= 5; i++) {
         try {
            Thread.sleep(1000); // 1-second delay
         } catch (InterruptedException e) {
            System.out.println(e.getMessage());
         }
         System.out.println(i);
      }
   }
}

public class SleepMethodExample {
   public static void main(String[] args) {
      SleepExample t = new SleepExample();
      t.start();
   }
}
```

4. yield() method.

Ans:-
```
class MyThread extends Thread {
   public void run() {
      for (int i = 0; i < 5; i++) {
         Thread.yield();
         System.out.println(Thread.currentThread().getName() + " is
running");
      }
   }
}
```

```
public class YieldMethodExample {
    public static void main(String[] args) {
        MyThread t1 = new MyThread();
        MyThread t2 = new MyThread();
        t1.start();
        t2.start();
    }
}
```

5. join() method.

Ans:-

```
class JoinExample extends Thread {
    public void run() {
        for (int i = 1; i <= 5; i++) {
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                System.out.println(e.getMessage());
            }
            System.out.println(i);
        }
    }
}

public class JoinMethodExample {
    public static void main(String[] args) {
        JoinExample t = new JoinExample();
        t.start();
        try {
            t.join();
        } catch (InterruptedException e) {
            System.out.println(e.getMessage());
        }
```

```
        System.out.println("Thread finished!");
    }
}
```

Q.4. Write an appropriate program by using the following method.

1. **wait() method.**
2. **notify() method.**
3. **notifyAll() method.**

E.g

Ans:-

```
class SharedResource {
    synchronized void waitMethod() {
        try {
            System.out.println("Waiting...");
            wait(); // Waits until notified
        } catch (InterruptedException e) {
            System.out.println(e.getMessage());
        }
        System.out.println("Resumed");
    }

    synchronized void notifyMethod() {
        System.out.println("Notifying...");
        notify(); // Wakes up a single thread
    }
}

public class WaitNotifyExample {
    public static void main(String[] args) {
        SharedResource resource = new SharedResource();
```

```
        Thread t1 = new Thread(resource::waitMethod);
        Thread t2 = new Thread(resource::notifyMethod);

        t1.start();
        try { Thread.sleep(1000); } catch (InterruptedException e) {}
        t2.start();
    }
}
```

Q.5. Write an appropriate program by using the following method.

1. isAlive() method.
2. suspend() method.

E.g

Ans:-

```
class MyThread extends Thread {
    public void run() {
        System.out.println("Thread is running...");
    }
}

public class IsAliveExample {
    public static void main(String[] args) {
        MyThread t = new MyThread();
        t.start();
        System.out.println("Is thread alive? " + t.isAlive());
    }
}
```

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:14

Objective: To learn about file handling concepts in Java.

Problem Definition:

Q.1. What is the concept of file handling in java? Explain with the help of an example.

Ans:- File handling in Java allows performing read, write, and manipulate operations on files. It uses classes from the java.io and java.nio packages.

E.g

```
import java.io.File;
import java.io.IOException;

public class FileHandlingExample {
   public static void main(String[] args) {
      try {
         File file = new File("example.txt");
         if (file.createNewFile()) {
            System.out.println("File created: " + file.getName());
         } else {
            System.out.println("File already exists.");
         }
      } catch (IOException e) {
         System.out.println("An error occurred.");
         e.printStackTrace();
      }
   }
}
```

Q.2. Explain the concept of stream class in java java.

Ans:- The Stream class is part of Java's I/O system to handle input and output of data. It represents a flow of data that can be either byte stream or character stream.

Key Classes:

- InputStream/OutputStream: For reading/writing bytes.
- Reader/Writer: For reading/writing characters.

Q.3. What is the File, FileWriter and FileReader class in Java? Explain with the help of a program example.

Ans:- File Class: Used to create and manage file/directory properties.
FileWriter Class: Used to write data to files.
FileReader Class: Used to read data from files.

E.g

```
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

public class FileReaderWriterExample {
   public static void main(String[] args) {
      try {
         FileWriter writer = new FileWriter("example.txt");
         writer.write("Hello, this is a FileWriter example.");
         writer.close();
         FileReader reader = new FileReader("example.txt");
         int character;
         while ((character = reader.read()) != -1) {
            System.out.print((char) character);
         }
```

```
            reader.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Q.4. What is the difference between the BufferedWriter and BufferedReader class in java?

Ans:-

BufferedWriter	BufferedReader
Writes text into a character stream efficiently.	Reads text from a character stream efficiently.
Used for writing.	Used for reading.
Example: `BufferedWriter bw = new BufferedWriter(new FileWriter("file.txt"));`	Example: `BufferedReader br = new BufferedReader(new FileReader("file.txt"));`

Q.5. Make an appropriate file handling program in java, where you have to write one paragraphs related to inheritance concept and one paragraph related to Polymorphism concept in text file, and then have to read all the content from text file to terminal/screen.

Ans:- import java.io.*;

```
public class FileParagraphExample {
    public static void main(String[] args) {
        try {
            BufferedWriter writer = new BufferedWriter(new
FileWriter("concepts.txt"));
            writer.write("Inheritance is a mechanism in Java where one
class acquires the properties of another class.");
            writer.newLine();
```

```
        writer.write("Polymorphism in Java allows a single action to
behave differently based on the object.");
        writer.close();

        BufferedReader reader = new BufferedReader(new
FileReader("concepts.txt"));
        String line;
        while ((line = reader.readLine()) != null) {
           System.out.println(line);
        }
        reader.close();
      } catch (IOException e) {
        e.printStackTrace();
      }
   }
}
```

Q.6. Make an appropriate file handling program in java, where you have to read minimum 5 employee information like (eid, ename,designation,salary and address) from user, and then have to write all information in a text file, and then have to print all this text file information on terminal/screen.

Ans:-

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

public class EmployeeInfo {
   public static void main(String[] args) {
      try {
        BufferedWriter writer = new BufferedWriter(new
FileWriter("employees.txt"));
        Scanner sc = new Scanner(System.in);
```

```
            for (int i = 0; i < 5; i++) {
                System.out.println("Enter Employee ID, Name, Designation,
Salary, Address:");
                String data = sc.nextLine();
                writer.write(data);
                writer.newLine();
            }
            writer.close();

            BufferedReader reader = new BufferedReader(new
FileReader("employees.txt"));
            String line;
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
            reader.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Output:-

```
D:\JealJava>java EmployeeInfo
Enter Employee ID, Name, Designation, Salary, Address:
1001
Enter Employee ID, Name, Designation, Salary, Address:
1002,Jeal,Cyber,15000,rfvvfhvbhuvb
Enter Employee ID, Name, Designation, Salary, Address:
cbyubrbvuv
Enter Employee ID, Name, Designation, Salary, Address:
fvdrdvb
Enter Employee ID, Name, Designation, Salary, Address:
gsrvb
1001
1002,Jeal,Cyber,15000,rfvvfhvbhuvb
cbyubrbvuv
fvdrdvb
gsrvb
```

Q.7. Make an appropriate file handling program in Java, where you have to read minimum 5 product related information like (pid, pname, qty, price) from user and then have to write all this information in a text file, and then have to print all this text file information on terminal / screen the following format like (pid, pname, qty , price , and total price). After printing all this information on screen you have to search individual product information by using pid or pname.

Ans:- import java.io.*;
import java.util.*;

```
public class ProductInfo {
   public static void main(String[] args) {
      try {
         Scanner sc = new Scanner(System.in);
         BufferedWriter writer = new BufferedWriter(new
FileWriter("products.txt"));
```

```
        for (int i = 0; i < 5; i++) {
            System.out.println("Enter Product ID, Name, Quantity,
Price:");
            String data = sc.nextLine();
            writer.write(data);
            writer.newLine();
        }
        writer.close();

        BufferedReader reader = new BufferedReader(new
FileReader("products.txt"));
        List<String> products = new ArrayList<>();
        String line;
        while ((line = reader.readLine()) != null) {
            products.add(line);
            System.out.println(line);
        }
        reader.close();

        System.out.println("Search by Product ID or Name:");
        String search = sc.nextLine();
        for (String product : products) {
            if (product.contains(search)) {
                System.out.println("Found: " + product);
            }
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
  }
}
```

Institute of Computer Technology

B. Tech. Computer Science and Engineering

Semester: III

Sub: Object-Oriented Programming

Course Code: 2CSE303

Practical Number:15

Objective: To learn about the JDBC concept in java.

Problem Definition:

Q.1. What is the concept of JDBC in java? Explain with examples.

Ans:- JDBC (Java Database Connectivity) is an API to connect and execute queries with a database. It allows Java applications to interact with relational databases.

E.g:-

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

public class JDBCBasics {
   public static void main(String[] args) {
      try {
         Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/testdb",
"root", "password");
         Statement stmt = con.createStatement();
         System.out.println("Connection successful!");
         con.close();
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```

Q.2. Explain different types of drivers in Java.

Ans:- Type 1: JDBC-ODBC Bridge Driver (uses ODBC for database interaction).

Type 2: Native-API Driver (uses database vendor's client-side libraries).

Type 3: Network Protocol Driver (uses middleware for communication).

Type 4: Thin Driver (pure Java driver interacting directly with the database).

Q.3. Explain the concept of Connection class, PreparedStatement class, and Statement class and the ResultSet class in java.

Ans:- Connection: Manages the connection with the database.

Statement: Executes SQL queries.

PreparedStatement: Precompiled SQL statements for efficiency.

ResultSet: Holds data retrieved from the database.

E.g:-

```
import java.sql.*;

public class JDBCClassesExample {
   public static void main(String[] args) {
      try {
         Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/testdb",
"root", "password");
         PreparedStatement pstmt = con.prepareStatement("SELECT *
FROM users WHERE id = ?");
```

```
            pstmt.setInt(1, 1);
            ResultSet rs = pstmt.executeQuery();
            while (rs.next()) {
               System.out.println("Name: " + rs.getString("name"));
            }
            con.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Q.4. Explain the concept of exception handling in the JDBC program.

Ans:- Exception handling in JDBC ensures smooth execution of the program in case of errors such as connection failure or invalid SQL commands.

E.g:-

```
try {
   Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/testdb",
"root", "password");
   // SQL operations
} catch (SQLException e) {
   System.out.println("SQL Exception: " + e.getMessage());
}
```

Q.5. What is database and table? Explain create, select, insert, modify, update and delete commands.

Ans:- CREATE: CREATE TABLE Emp (emp_id INT, emp_name VARCHAR(50));

INSERT: INSERT INTO Emp VALUES (1, 'John');

SELECT: SELECT * FROM Emp;

UPDATE: UPDATE Emp SET emp_name = 'Mike' WHERE emp_id = 1;

DELETE: DELETE FROM Emp WHERE emp_id = 1;

Q.6. Design a system, where you have to create one database with the name of "Employee", and then you have to create one table with the name of "Emp" with the following column field details like (emp_id, emp_name, designation, salary, address, and contact number). After creating the database and table, you have to insert a minimum of 5 employee records in the table, and then you have to display all the table records on terminal / screen. and then you have to search individual employee records by using emp_id or by emp_name. So, for the fulfillment purpose of the above said requirement write an appropriate JDBC program in java.

Ans:- import java.sql.*;

```
public class EmployeeDatabase {
   public static void main(String[] args) {
      try {
         Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/",
"root", "jeal259372");
         Statement stmt = con.createStatement();
```

```
        // Create Database and Table
        stmt.execute("CREATE DATABASE IF NOT EXISTS Employee");
        stmt.execute("USE Employee");
        stmt.execute("CREATE TABLE IF NOT EXISTS Emp (emp_id INT
PRIMARY KEY, emp_name VARCHAR(50), designation VARCHAR(50),
salary DOUBLE, address VARCHAR(100), contact_number
VARCHAR(15))");

        // Insert Records
        stmt.execute("INSERT INTO Emp VALUES (1, 'John', 'Manager',
50000, 'NY', '1234567890')");
        stmt.execute("INSERT INTO Emp VALUES (2, 'Jane',
'Developer', 45000, 'LA', '0987654321')");

        // Display Records
        ResultSet rs = stmt.executeQuery("SELECT * FROM Emp");
        while (rs.next()) {
            System.out.println(rs.getInt("emp_id") + " " +
rs.getString("emp_name"));
        }

        // Search by ID or Name
        PreparedStatement pstmt = con.prepareStatement("SELECT *
FROM Emp WHERE emp_id = ? OR emp_name = ?");
        pstmt.setInt(1, 1);
        pstmt.setString(2, "Jane");
        ResultSet searchResult = pstmt.executeQuery();
        while (searchResult.next()) {
            System.out.println("Found: " +
searchResult.getString("emp_name"));
        }
```

```
            con.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Q.7. Write an appropriate JDBC program in java, where you have to check if the connection is created or not with the database.

Ans:-

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

public class StudentDatabase {
    public static void main(String[] args) {
        try {
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/",
"root", "password");
            Statement stmt = con.createStatement();

            // Create Database and Table
            stmt.execute("CREATE DATABASE IF NOT EXISTS Student");
            stmt.execute("USE Student");
            stmt.execute("CREATE TABLE IF NOT EXISTS Stud (stud_id INT
PRIMARY KEY, name VARCHAR(50), course VARCHAR(50), branch
VARCHAR(50), fee DOUBLE, contact_number VARCHAR(15))");

            // Insert Records
            stmt.execute("INSERT INTO Stud VALUES (1, 'Alice', 'B.Tech',
'CSE', 50000, '1234567890')");
```

```
        stmt.execute("INSERT INTO Stud VALUES (2, 'Bob', 'B.Sc', 'IT',
45000, '0987654321')");

        // Display Records
        ResultSet rs = stmt.executeQuery("SELECT * FROM Stud");
        while (rs.next()) {
            System.out.println(rs.getInt("stud_id") + " " +
rs.getString("name"));
        }

        // Search Records
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter Student ID or Name:");
        String search = sc.nextLine();
        PreparedStatement pstmt = con.prepareStatement("SELECT *
FROM Stud WHERE stud_id = ? OR name = ?");
        pstmt.setString(1, search);
        pstmt.setString(2, search);
        ResultSet searchResult = pstmt.executeQuery();
        while (searchResult.next()) {
            System.out.println("Found: " +
searchResult.getString("name"));
        }

        // Modify Records
        System.out.println("Enter ID to Update Fee:");
        int id = sc.nextInt();
        System.out.println("Enter New Fee:");
        double fee = sc.nextDouble();
        pstmt = con.prepareStatement("UPDATE Stud SET fee = ?
WHERE stud_id = ?");
```

```
            pstmt.setDouble(1, fee);
            pstmt.setInt(2, id);
            pstmt.executeUpdate();
            System.out.println("Record Updated!");

            // Delete Records
            System.out.println("Enter ID to Delete:");
            int delId = sc.nextInt();
            pstmt = con.prepareStatement("DELETE FROM Stud WHERE
stud_id = ?");
            pstmt.setInt(1, delId);
            pstmt.executeUpdate();
            System.out.println("Record Deleted!");

            con.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```