1. Write a java program to find the Fibonacci series using recursive and non-recursive functions.

public class FibonacciRecursive {

public static void main(String[] args) {

int n = 10; // Number of terms to display

System.out.print("Fibonacci Series using recursion: ");

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

System.out.print(fibonacci(i) + " ");

}

}

public static int fibonacci(int n) {

if (n <= 1) {

return n;

}

return fibonacci(n - 1) + fibonacci(n - 2);

}

}

\_\_\_

public class FibonacciNonRecursive {

public static void main(String[] args) {

int n = 10; // Number of terms to display

System.out.print("Fibonacci Series using iteration: ");

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

System.out.print(fibonacci(i) + " ");

}

}

public static int fibonacci(int n) {

if (n <= 1) {

return n;

}

int a = 0, b = 1, c;

for (int i = 2; i <= n; i++) {

c = a + b;

a = b;

b = c;

}

return b;

}

}

1. Write a java program to multiply two given matrice

import java.util.Scanner;

public class MatrixMultiplication {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

// Input the dimensions of the first matrix

System.out.print("Enter the number of rows in the first matrix: ");

int rows1 = sc.nextInt();

System.out.print("Enter the number of columns in the first matrix: ");

int cols1 = sc.nextInt();

// Input the dimensions of the second matrix

System.out.print("Enter the number of rows in the second matrix: ");

int rows2 = sc.nextInt();

System.out.print("Enter the number of columns in the second matrix: ");

int cols2 = sc.nextInt();

// Check if matrix multiplication is possible

if (cols1 != rows2) {

System.out.println("Matrix multiplication not possible! The number of columns in the first matrix must equal the number of rows in the second matrix.");

return;

}

// Input the elements of the first matrix

int[][] matrix1 = new int[rows1][cols1];

System.out.println("Enter the elements of the first matrix:");

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols1; j++) {

matrix1[i][j] = sc.nextInt();

}

}

// Input the elements of the second matrix

int[][] matrix2 = new int[rows2][cols2];

System.out.println("Enter the elements of the second matrix:");

for (int i = 0; i < rows2; i++) {

for (int j = 0; j < cols2; j++) {

matrix2[i][j] = sc.nextInt();

}

}

// Multiply the matrices

int[][] result = new int[rows1][cols2];

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols2; j++) {

for (int k = 0; k < cols1; k++) {

result[i][j] += matrix1[i][k] \* matrix2[k][j];

}

}

}

// Display the result

System.out.println("Result of matrix multiplication:");

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols2; j++) {

System.out.print(result[i][j] + " ");

}

System.out.println();

}

sc.close();

}

}

1. Write a java program to display the employee details using Scanner class.

import java.util.Scanner

class Employee {

private String name;

private int id;

private String department;

private double salary;

public Employee(String name, int id, String department, double salary) {

this.name = name;

this.id = id;

this.department = department;

this.salary = salary;

}

// Method to display employee details

public void displayDetails() {

System.out.println("Employee ID: " + id);

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

System.out.println("Department: " + department);

System.out.println("Salary: " + salary);

}

}

public class EmployeeDetails {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Input employee details

System.out.print("Enter Employee ID: ");

int id = scanner.nextInt();

scanner.nextLine(); // Consume the newline character

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

String name = scanner.nextLine();

System.out.print("Enter Employee Department: ");

String department = scanner.nextLine();

System.out.print("Enter Employee Salary: ");

double salary = scanner.nextDouble();

// Create an Employee object

Employee employee = new Employee(name, id, department, salary);

// Display employee details

System.out.println("\nEmployee Details:");

employee.displayDetails();

// Close the scanner

scanner.close();

}

}

1. Write a Package MCA which has one class Student. Accept student detail through the parameterized constructor. Write display () method to display details. Create a main class which will use package and calculate total marks and percentage.

package MCA;

public class Student {

private String name;

private int rollNo;

private int[] marks;

// Parameterized constructor

public Student(String name, int rollNo, int[] marks) {

this.name = name;

this.rollNo = rollNo;

this.marks = marks;

}

// Method to display student details

public void display() {

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

System.out.println("Roll No: " + rollNo);

System.out.print("Marks: ");

for (int mark : marks) {

System.out.print(mark + " ");

}

System.out.println();

}

// Getter for marks

public int[] getMarks() {

return marks;

}

}

// File: MainClass.java

import MCA.Student;

public class MainClass {

public static void main(String[] args) {

// Sample student details

String name = "John Doe";

int rollNo = 101;

int[] marks = {85, 90, 78, 92, 88}; // Example marks

// Create a Student object

Student student = new Student(name, rollNo, marks);

// Display student details

student.display();

// Calculate total marks and percentage

int totalMarks = 0;

for (int mark : marks) {

totalMarks += mark;

}

double percentage = (double) totalMarks / marks.length;

// Display total marks and percentage

System.out.println("Total Marks: " + totalMarks);

System.out.println("Percentage: " + percentage + "%");

}

}

1. Write an applet program that displays a simple message

import java.awt.\*;

import java.applet.\*;

/\* <applet code="FirstApplet" width=400 height=300></applet> \*/

public class FirstApplet extends Applet {

public void paint(Graphics g) {

g.setColor(Color.blue);

Font font = new Font("Arial", Font.BOLD, 16);

g.setFont(font);

g.drawString("This is My First Applet", 60, 110);

}

}

6. A) Write a Java program compute factorial value using Applet.

import java.awt.\*;

import java.applet.\*;

import java.awt.event.\*;

/\*<applet code="FactorialApplet" width=500 height=250>

</applet>\*/

public class FactorialApplet extends Applet implements ActionListener {

Label L1,L2;

TextField T1,T2;

Button B1;

public void init() {

L1=new Label("Enter any Number : ");

add(L1);

T1=new TextField(10);

add(T1);

L2=new Label("Factorial of Num : ");

add(L2);

T2=new TextField(10);

add(T2);

B1=new Button("Compute");

add(B1);

B1.addActionListener(this);

}

public void actionPerformed(ActionEvent e) {

if(e.getSource()==B1)

{

int value=Integer.parseInt(T1.getText());

int fact=factorial(value);

T2.setText(String.valueOf(fact));

}

}

int factorial(int n) {

if(n==0)

return 1;

else

return n\*factorial(n-1);

}

}

B) Write a program for passing parameters using Applet

import java.applet.Applet;

import java.awt.Graphics;

/\*

<applet code="AppletExample" width=300 height=300>

<param name="message" value="Hello, Welcome to Applet Programming!">

<param name="name" value="Varun">

</applet>

\*/

public class AppletExample extends Applet {

String message;

String name;

// The init() method is called when the applet is first loaded

public void init() {

// Retrieve parameters from HTML

message = getParameter("message");

name = getParameter("name");

// Set default values if parameters are not provided

if (message == null) {

message = "No message provided!";

}

if (name == null) {

name = "Guest";

}

}

// The paint() method is called to draw the content of the applet

public void paint(Graphics g) {

g.drawString(message, 20, 50);

g.drawString("Name: " + name, 20, 80);

}

}

7. Write a java program for handling Mouse events and Key events.

**import java.awt.Color;**

**import java.awt.Graphics;**

**import java.awt.event.KeyEvent;**

**import java.awt.event.KeyListener;**

**import java.awt.event.MouseEvent;**

**import java.awt.event.MouseListener;**

**import java.awt.event.MouseMotionListener;**

**import javax.swing.JApplet;**

**public class EventHandlingExample extends JApplet implements MouseListener, MouseMotionListener, KeyListener {**

**private int mouseX = 0, mouseY = 0; // Coordinates for mouse events**

**private String keyPressed = ""; // Stores the last key pressed**

**@Override**

**public void init() {**

**// Register the applet to listen to mouse and key events**

**addMouseListener(this);**

**addMouseMotionListener(this);**

**addKeyListener(this);**

**setFocusable(true); // Make sure the applet is focused to receive key events**

**requestFocusInWindow();**

**}**

**@Override**

**public void paint(Graphics g) {**

**super.paint(g);**

**// Clear the background**

**g.setColor(Color.WHITE);**

**g.fillRect(0, 0, getWidth(), getHeight());**

**// Draw the mouse coordinates**

**g.setColor(Color.BLUE);**

**g.drawString("Mouse at: (" + mouseX + ", " + mouseY + ")", 10, 20);**

**// Draw the last key pressed**

**g.setColor(Color.RED);**

**g.drawString("Last Key Pressed: " + keyPressed, 10, 40);**

**}**

**// MouseListener methods**

**@Override**

**public void mouseClicked(MouseEvent e) {**

**mouseX = e.getX();**

**mouseY = e.getY();**

**repaint();**

**}**

**@Override**

**public void mousePressed(MouseEvent e) {**

**mouseX = e.getX();**

**mouseY = e.getY();**

**repaint();**

**}**

**@Override**

**public void mouseReleased(MouseEvent e) {**

**mouseX = e.getX();**

**mouseY = e.getY();**

**repaint();**

**}**

**@Override**

**public void mouseEntered(MouseEvent e) {**

**mouseX = e.getX();**

**mouseY = e.getY();**

**repaint();**

**}**

**@Override**

**public void mouseExited(MouseEvent e) {**

**mouseX = -1;**

**mouseY = -1;**

**repaint();**

**}**

**// MouseMotionListener methods**

**@Override**

**public void mouseDragged(MouseEvent e) {**

**mouseX = e.getX();**

**mouseY = e.getY();**

**repaint();**

**}**

**@Override**

**public void mouseMoved(MouseEvent e) {**

**mouseX = e.getX();**

**mouseY = e.getY();**

**repaint();**

**}**

**// KeyListener methods**

**@Override**

**public void keyTyped(KeyEvent e) {**

**keyPressed = String.valueOf(e.getKeyChar());**

**repaint();**

**}**

**@Override**

**public void keyPressed(KeyEvent e) {**

**keyPressed = KeyEvent.getKeyText(e.getKeyCode());**

**repaint();**

**}**

**@Override**

**public void keyReleased(KeyEvent e) {**

**keyPressed = "";**

**repaint();**

**}**

**}**

9. Write a java program that connects to a database using JDBC connection.

import java.sql.\*;

import javax.sql.\*;

public class Pmg\_connect

{

public static void main(String[] args)

{

String url = "jdbc:mysql://localhost:3306/Pmg9\_connect?useSSL=false";

String username = "root";

String password = "root";

try

{

Connection con = DriverManager.getConnection(url, username, password);

System.out.println("Connected to MySQL database successfully!");

con.close();

}

catch (SQLException e)

{

e.printStackTrace();

}

}

}

10. Write a java program to connect to a database using JDBC and insert values into it

**import java.sql.Connection;**

**import java.sql.DriverManager;**

**import java.sql.PreparedStatement;**

**import java.sql.SQLException;**

**public class JDBCExample {**

**// JDBC URL, username, and password of MySQL server**

**static final String DB\_URL = "jdbc:mysql://localhost:3306/yourDatabase"; // Update "yourDatabase" with your actual database name**

**static final String USER = "yourUsername"; // Update with your MySQL username**

**static final String PASS = "yourPassword"; // Update with your MySQL password**

**// JDBC Connection and PreparedStatement**

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

**Connection conn = null;**

**PreparedStatement pstmt = null;**

**try {**

**// Step 1: Register JDBC driver**

**Class.forName("com.mysql.cj.jdbc.Driver");**

**// Step 2: Open a connection**

**System.out.println("Connecting to database...");**

**conn = DriverManager.getConnection(DB\_URL, USER, PASS);**

**// Step 3: Create SQL query**

**String sql = "INSERT INTO students (id, name, age) VALUES (?, ?, ?)";**

**// Step 4: Prepare statement**

**pstmt = conn.prepareStatement(sql);**

**pstmt.setInt(1, 1); // For example, inserting student ID = 1**

**pstmt.setString(2, "Varun"); // Inserting name = "Varun"**

**pstmt.setInt(3, 22); // Inserting age = 22**

**// Step 5: Execute the query**

**int rowsInserted = pstmt.executeUpdate();**

**if (rowsInserted > 0) {**

**System.out.println("A new record was inserted successfully!");**

**}**

**} catch (SQLException se) {**

**// Handle errors for JDBC**

**se.printStackTrace();**

**} catch (Exception e) {**

**// Handle errors for Class.forName**

**e.printStackTrace();**

**} finally {**

**// Step 6: Close resources**

**try {**

**if (pstmt != null) pstmt.close();**

**if (conn != null) conn.close();**

**} catch (SQLException se) {**

**se.printStackTrace();**

**}**

**}**

**}**

**}**

Ensure you have a table in your database. Here’s a sample SQL statement to create the students table:

CREATE TABLE students (

id INT PRIMARY KEY,

name VARCHAR(50),

age INT

);