AMITY UNIVERSITY MADHYA PRADESH, GWALIOR

ADVANCED JAVA PROGRAMMING

LAB FILE

(CSE 524)

Submitted To:
DR. Rajeev Goyal

Submitted By:
Shailendra Mourya
B. Tech (CSE), 5th sem
A60205222016 “A1”

INDEX

S. No.	Program/Objective	Date	Remark/Sign
1.	To install and configure Java Development Kit (JDK) version 23 on the system.		
2.	To install and configure Java Development Kit (JDK) version 8 on the system.		
3.	To install and configure Eclipse IDE on the system.		
4.	Implement a Remote Method Invocation (RMI) system in Java that allows two-way communication between the client and server for a simple addition operation.		
5.	Develop a graphical user interface (GUI) application using Java Swing for a basic calculator that performs addition, subtraction, and multiplication operations.		
6.	Design and implement a calculator application using Java Swing that supports basic arithmetic operations such as addition, subtraction, multiplication, and division.		
7.	Develop a simple notepad application using Java Swing.		
8.	Demonstrate JDBC in a Java application by performing CRUD operations on a MySQL database. The application should include: 1. Insertion of a record using a Statement. 2. Update of a record using a Statement. 3. Deletion of a record using a Statement. 4. Retrieval of records using a Statement. 5. Insertion of a record using a PreparedStatement with dynamic input from the user.		
9.	Create a simple Servlet in Java that displays a "Hello World" message when accessed.		
10.	Create a web-based student login system using servlets and JSP.		
11.	Create a simple web application to calculate the average of two numbers using the MVC architecture in servlets and JSP.		
12.	Demonstrate the use of Servlet Session Management		
13.	EJB		

QUES-1 To install and configure Java Development Kit (JDK) version 23 on the system.

ANS

Step 1: Download JDK 23

1. Visit the [official Oracle Java SE Downloads page]() or a trusted source for JDK 23.
2. Select the appropriate version for your operating system (Windows, macOS, or Linux) and architecture (e.g., x64 or ARM).
3. Download the installer or compressed package.

Linux **macOS** **Windows**

Product/file description	File size	Download
x64 Compressed Archive	228.76 MB	https://download.oracle.com/java/23/latest/jdk-23_windows-x64_bin.zip (sha256)
x64 Installer	205.26 MB	https://download.oracle.com/java/23/latest/jdk-23_windows-x64_bin.exe (sha256)
x64 MSI Installer	204.00 MB	https://download.oracle.com/java/23/latest/jdk-23_windows-x64_bin.msi (sha256)

Step 2: Install JDK 23

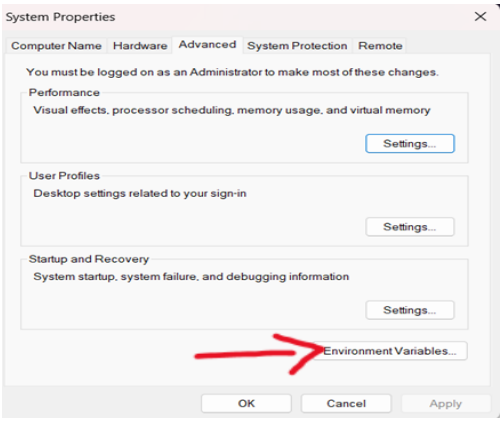
For Windows:

1. Run the downloaded .exe installer.
2. Follow the installation wizard:
    - Accept the license agreement.
    - Choose the installation directory (default: C:\Program Files\Java\jdk-23).
    - Complete the installation.

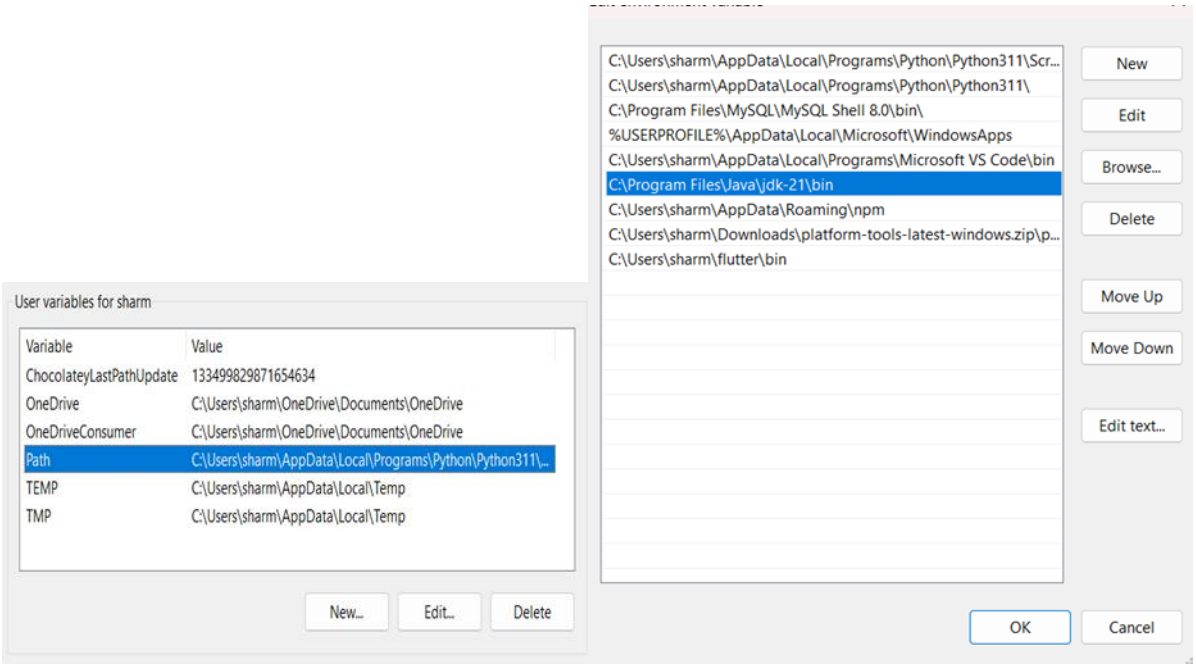
Step 3: Configure Environment Variables

For Windows:

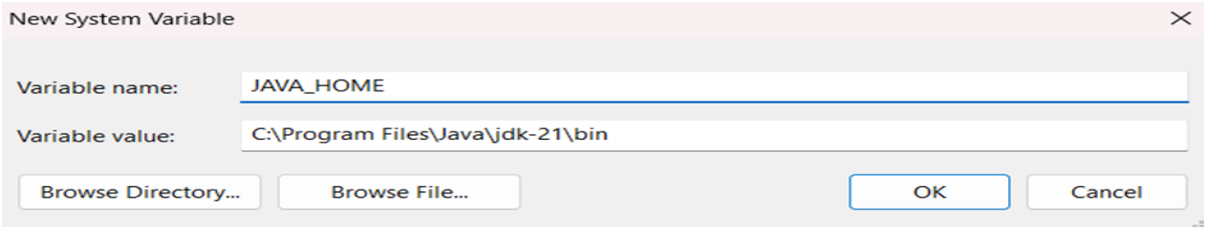
1. Open **System Properties**:
    - Search for "Environment Variables" in the Start menu.



2. Add a new **System Variable**:
    - Variable Name: JAVA_HOME
    - Variable Value: Path to the JDK installation (e.g., C:\Program Files\Java\jdk-23).



3. Update the Path variable:
    - Add %JAVA_HOME%\bin to the Path



Step 4: Verify Installation

Open a terminal or command prompt and run:

java -version

```
PS C:\Users\asus> java -version
java version "23.0.1" 2024-10-15
Java(TM) SE Runtime Environment (build 23.0.1+11-39)
Java HotSpot(TM) 64-Bit Server VM (build 23.0.1+11-39, mixed mode, sharing)
PS C:\Users\asus>
```

This should display the JDK 23 version information.

Test javac

javac -version

```
PS C:\Users\asus> javac -version
javac 23.0.1
```

QUES 2 - To install and configure Java Development Kit (JDK) version 8 on the system.

ANS -

Step 1: Download JDK 8

1. Go to the [official Oracle Java SE 8 Downloads page](#).
2. Select the appropriate JDK 8 version for your operating system (Windows, macOS, or Linux) and architecture (e.g., x64 or x86).
3. Download the installer or compressed package.

Linux **macOS** **Windows**

Product/file description	File size	Download
x64 Compressed Archive	228.76 MB	https://download.oracle.com/java/23/latest/jdk-23_windows-x64_bin.zip (sha256)
x64 Installer	205.26 MB	https://download.oracle.com/java/23/latest/jdk-23_windows-x64_bin.exe (sha256)
x64 MSI Installer	204.00 MB	https://download.oracle.com/java/23/latest/jdk-23_windows-x64_bin.msi (sha256)

Step 2: Install JDK 8

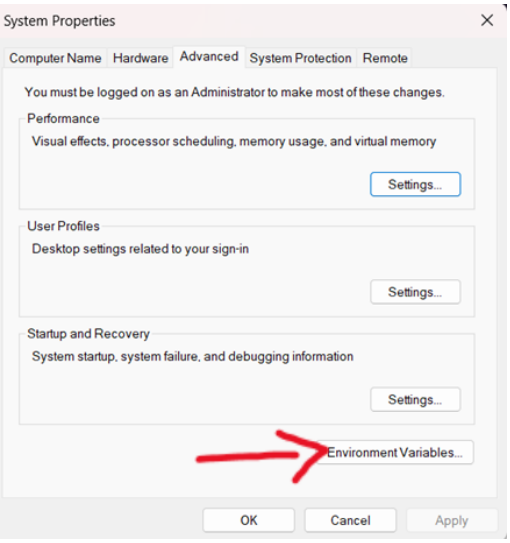
For Windows:

1. Run the downloaded .exe installer.
2. Follow the installation wizard:
   - Accept the license agreement.
   - Choose the installation directory (default: C:\Program Files\Java\jdk1.8.x_xx).
   - Complete the installation.

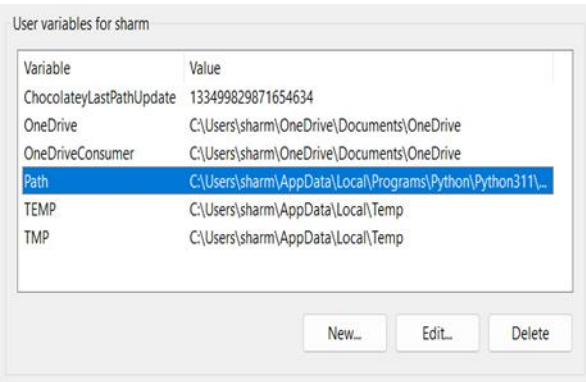
Step 3: Configure Environment Variables

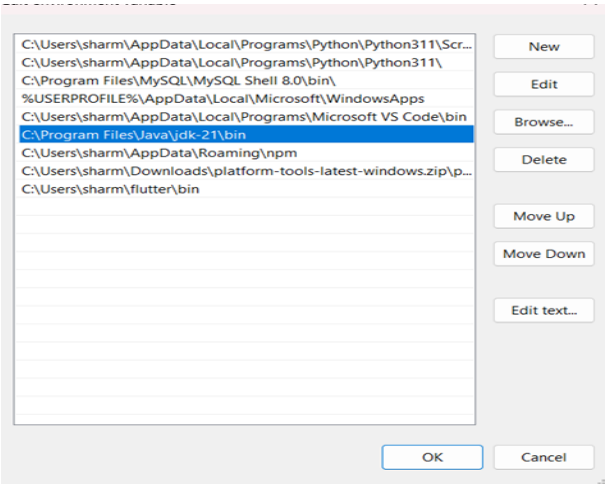
For Windows:

1. Open **System Properties**:
   - Search for "Environment Variables" in the Start menu.

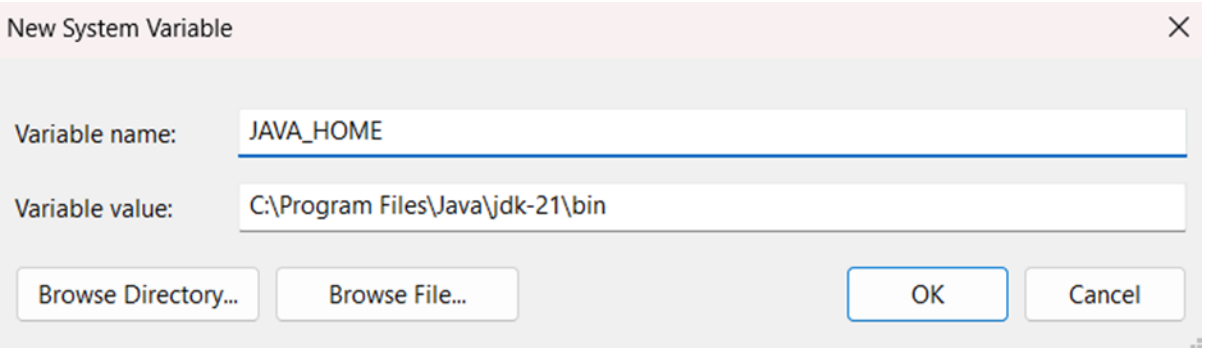


2. Add a new **System Variable**:
    - Variable Name: JAVA_HOME
    - Variable Value: Path to the JDK installation (e.g., C:\Program Files\Java\jdk1.8.x_xx).





3. Update the Path variable:
    - Add %JAVA_HOME%\bin to the Path.



Step 4: Verify Installation

Open a terminal or command prompt and run:

java -version

```
PS C:\Users\asus> java -version
java version "1.8.0_431"
Java(TM) SE Runtime Environment (build 1.8.0_431-b10)
Java HotSpot(TM) 64-Bit Server VM (build 25.431-b10, mixed mode)
PS C:\Users\asus>
```

This should display the JDK 8 version information.

Test javac:

javac -version

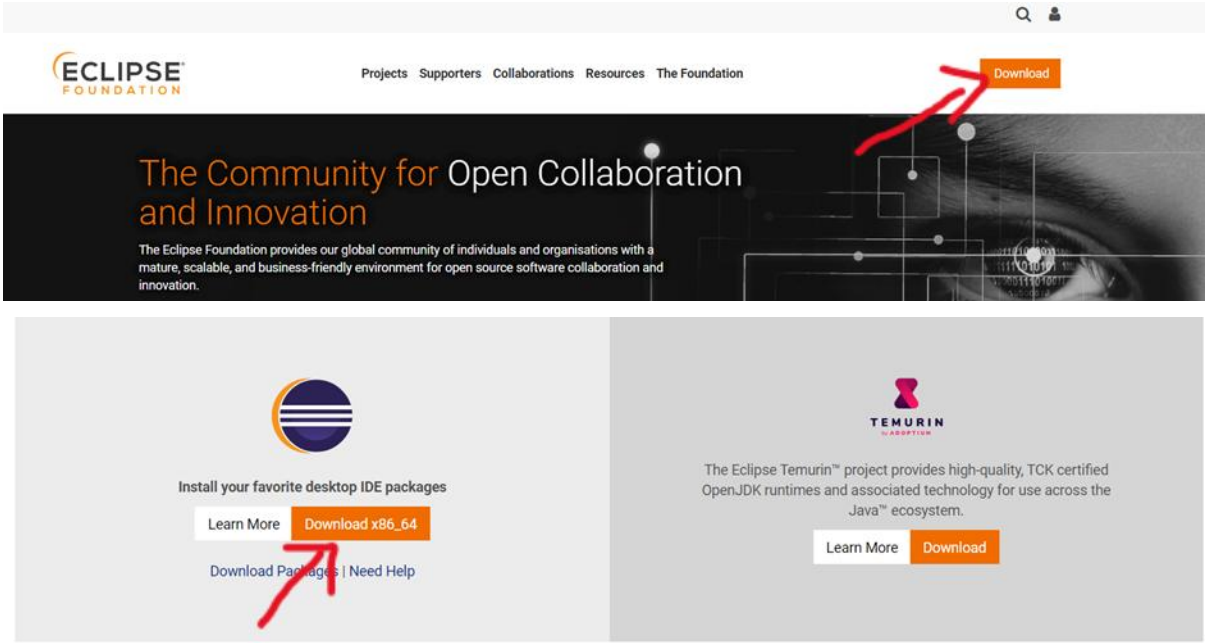
```
PS C:\Users\asus> javac -version
javac 1.8.0_431
```

QUES 3- To install and configure Eclipse IDE on the system.

ANS-

Step 1: Download Eclipse IDE

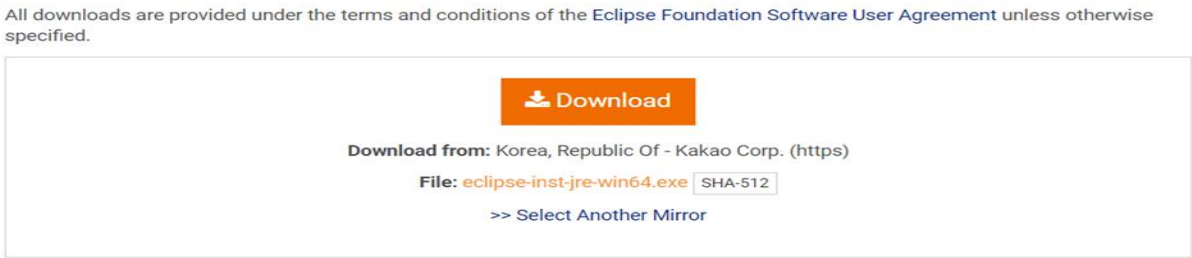
1. Visit the official Eclipse Downloads page.
2. Select the appropriate Eclipse package based on your development needs:
    - For Java developers: "Eclipse IDE for Enterprise Java and Web Developers."
    - For other purposes (e.g., C/C++, Web development): Select the relevant package.
3. Download the installer suitable for your operating system.



Step 2: Install Eclipse IDE

For Windows:

1. Extract the downloaded .zip file to a preferred location (e.g., C:\Program Files\Eclipse).
2. Open the extracted folder and double-click eclipse.exe to launch Eclipse.



Step 3: Configure Eclipse IDE

1. **Select a Workspace Directory**:
   - On first launch, Eclipse will prompt you to select a workspace directory. This is where your projects will be stored.
   - Choose a location or accept the default path.
2. **Set Up JDK in Eclipse**:
   - Go to **Window** > **Preferences** (or **Eclipse** > **Preferences** on macOS).
   - Navigate to **Java** > **Installed JREs**.
   - Click **Add**, select Standard VM, and browse to your JDK installation directory (e.g., C:\Program Files\Java\jdk-XX).

Step 4: Verify Eclipse Setup

1. **Create a New Java EE Project**:
   - Go to **File** > **New** > **Dynamic Web Project**.
   - Enter the project name and click **Finish**.
2. **Run a Simple Web Application**:
   - Create a basic servlet or JSP file.
   - Add a server (e.g., Apache Tomcat):
     - Go to **Window** > **Show View** > **Servers**.
     - Right-click the **Servers** view, select **New** > **Server**.
     - Choose a server runtime environment (e.g., Tomcat) and configure it.
   - Start the server and run the application.

OUTPUT -:

JSP FILE (index.jsp)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>
<!DOCTYPE html>
```

```
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<h1>Hello World</h1>
</body>
</html>
```

Hello World

Ques 4 - Implement a Remote Method Invocation (RMI) system in Java that allows two-way communication between the client and server for a simple addition operation.

ANS -

1). Interface

```
import java.rmi.Remote;

public interface CalcI extends Remote{

public int add(int x, int y)throws Exception;
}
```

2). Implementation of Interface

```
import java.rmi.server.UnicastRemoteObject;
public class CalcClass extends UnicastRemoteObject implements CalcI{
public CalcClass()throws Exception{
super();
}
public int add(int x, int y)
{
return x+y;
}
}
```

3). Server

```
import java.rmi.Naming;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

public class Server {

public static void main(String[] args)throws Exception{
   Registry registry = LocateRegistry.createRegistry(5099);
CalcClass obj = new CalcClass();
registry.rebind("calc",obj);
System.out.println("Server started");
}
}
```

4). Client

```
import java.rmi.Naming;
import java.rmi.RemoteException;
public class Client{

public static void main(String[] args) throws Exception{
try{
CalcI obj = (CalcI) Naming.lookup("rmi://localhost:5099/calc");
int n = obj.add(5,7);
System.out.println(n);
}catch (RemoteException e)
{
      System.out.println(e);
}catch (Exception e)
{
      System.out.print(e);
     }
}
}
```

Output:

```
sharma@sharma:/mnt/c/Users/sharma/sharma/coding/java/server$ java Server
Server Started
```

```
sharma@sharma:/mnt/c/Users/sharma/sharma/coding/java/server$ java Client
Addition of 10 and 20 is 30
```

QUES 5 - Develop a graphical user interface (GUI) application using Java Swing for a basic calculator that performs addition, subtraction, and multiplication operations.

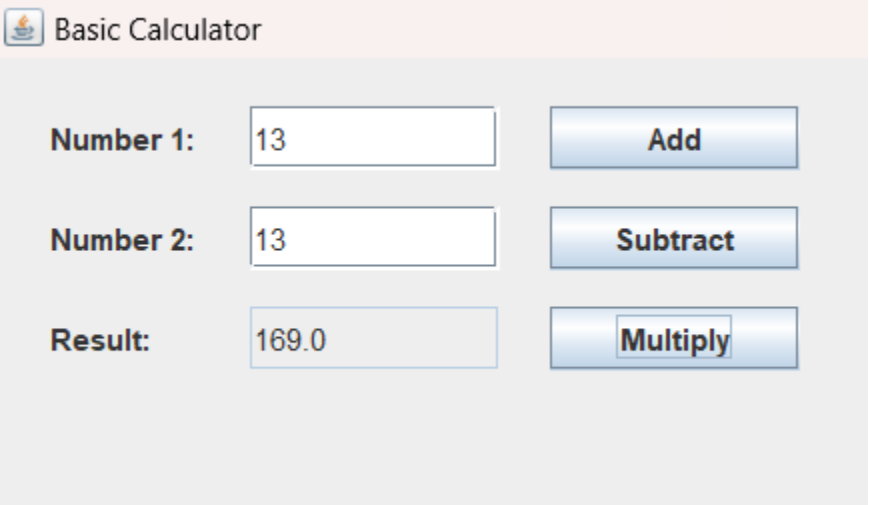
ANS -

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

public class BasicCalculator {
    public static void main(String[] args) {
        // Create the main frame
        JFrame frame = new JFrame("Basic Calculator");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 200);
        frame.setLayout(null);

        // Create input fields and labels
        JLabel label1 = new JLabel("Number 1:");
        label1.setBounds(20, 20, 80, 25);
        JTextField num1Field = new JTextField();
        num1Field.setBounds(100, 20, 100, 25);

        JLabel label2 = new JLabel("Number 2:");
        label2.setBounds(20, 60, 80, 25);
        JTextField num2Field = new JTextField();
        num2Field.setBounds(100, 60, 100, 25);

        JLabel resultLabel = new JLabel("Result:");
        resultLabel.setBounds(20, 100, 80, 25);
        JTextField resultField = new JTextField();
        resultField.setBounds(100, 100, 100, 25);
        resultField.setEditable(false);

        // Create buttons for operations
        JButton addButton = new JButton("Add");
        addButton.setBounds(220, 20, 100, 25);
        JButton subtractButton = new JButton("Subtract");
        subtractButton.setBounds(220, 60, 100, 25);
        JButton multiplyButton = new JButton("Multiply");
        multiplyButton.setBounds(220, 100, 100, 25);

        // Add action listeners for buttons
        addButton.addActionListener(new ActionListener() {
```

```
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    double num1 = Double.parseDouble(num1Field.getText());
                    double num2 = Double.parseDouble(num2Field.getText());
                    resultField.setText(String.valueOf(num1 + num2));
                } catch (NumberFormatException ex) {
                    JOptionPane.showMessageDialog(frame, "Please enter valid numbers.", "Error",
JOptionPane.ERROR_MESSAGE);
                }
            }
        });

        subtractButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    double num1 = Double.parseDouble(num1Field.getText());
                    double num2 = Double.parseDouble(num2Field.getText());
                    resultField.setText(String.valueOf(num1 - num2));
                } catch (NumberFormatException ex) {
                    JOptionPane.showMessageDialog(frame, "Please enter valid numbers.", "Error",
JOptionPane.ERROR_MESSAGE);
                }
            }
        });

        multiplyButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    double num1 = Double.parseDouble(num1Field.getText());
                    double num2 = Double.parseDouble(num2Field.getText());
                    resultField.setText(String.valueOf(num1 * num2));
                } catch (NumberFormatException ex) {
                    JOptionPane.showMessageDialog(frame, "Please enter valid numbers.", "Error",
JOptionPane.ERROR_MESSAGE);
                }
            }
        });

        // Add components to the frame
        frame.add(label1);
```

```
        frame.add(num1Field);
        frame.add(label2);
        frame.add(num2Field);
        frame.add(resultLabel);
        frame.add(resultField);
        frame.add(addButton);
        frame.add(subtractButton);
        frame.add(multiplyButton);

        // Make the frame visible
        frame.setVisible(true);
    }
}
```

OUTPUT -:



QUES 6 - Design and implement a calculator application using Java Swing that supports basic arithmetic operations such as addition, subtraction, multiplication, and division.

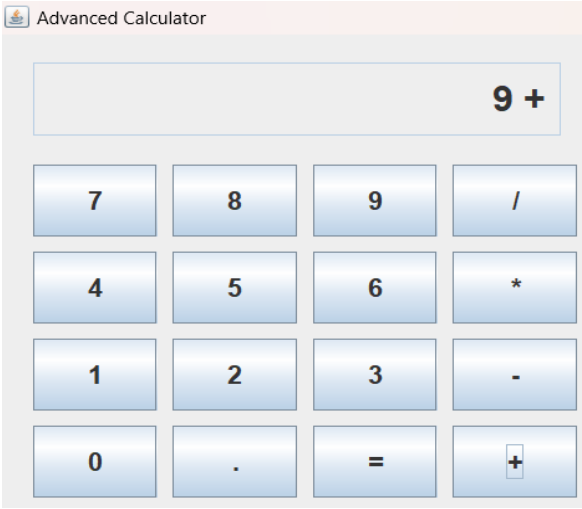
ANS –

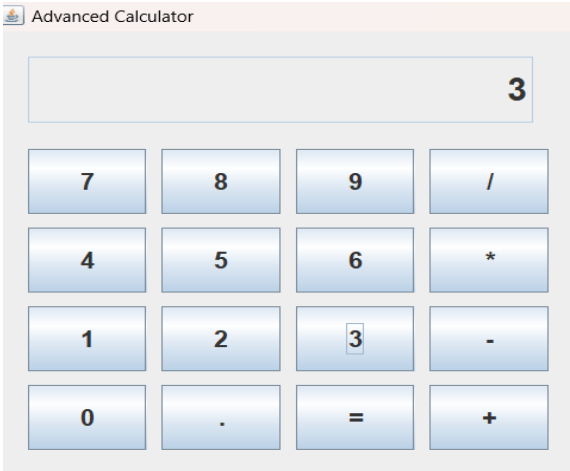
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class AdvancedCalculator {
    public static void main(String[] args) {
        // Create the main frame
        JFrame frame = new JFrame("Advanced Calculator");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 500);
        frame.setLayout(null);

        // Create a display area
        JTextField displayField = new JTextField();
        displayField.setBounds(20, 20, 340, 50);
        displayField.setFont(new Font("Arial", Font.BOLD, 24));
        displayField.setHorizontalAlignment(JTextField.RIGHT);
        displayField.setEditable(false);
        frame.add(displayField);

        // Button labels for calculator
        String[] buttonLabels = {
            "7", "8", "9", "/",
            "4", "5", "6", "*",
            "1", "2", "3", "-",
            "0", ".", "=", "+"
        };

        // Create buttons
        JButton[] buttons = new JButton[buttonLabels.length];
        int x = 20, y = 90;
        for (int i = 0; i < buttonLabels.length; i++) {
            buttons[i] = new JButton(buttonLabels[i]);
            buttons[i].setBounds(x, y, 80, 50);
            buttons[i].setFont(new Font("Arial", Font.BOLD, 18));
```

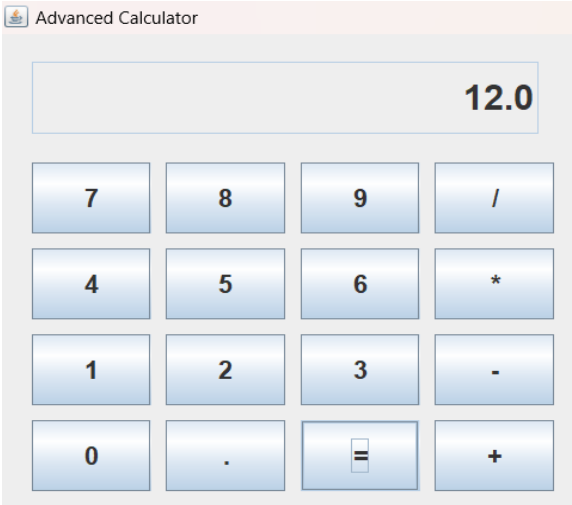
```
    frame.add(buttons[i]);
    x += 90;
    if ((i + 1) % 4 == 0) {
        x = 20;
        y += 60;
    }
}

// Action listeners for calculator logic
final StringBuilder currentInput = new StringBuilder();
final double[] result = {0};
final String[] operator = {""};
final boolean[] isResultDisplayed = {false};

for (JButton button : buttons) {
    button.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            String text = button.getText();

            if (isResultDisplayed[0] && "0123456789.".contains(text)) {
                // Clear display if a number is entered after result
                currentInput.setLength(0);
                isResultDisplayed[0] = false;
            }

            if ("0123456789.".contains(text)) {
                currentInput.append(text);
                displayField.setText(currentInput.toString());
            } else if ("+-*/".contains(text)) {
                if (currentInput.length() > 0) {
                    if (!isResultDisplayed[0]) {
                        result[0] = Double.parseDouble(currentInput.toString());
                    }
                    currentInput.setLength(0);
                }
                operator[0] = text;
                displayField.setText(displayField.getText() + " " + operator[0] + " ");
                isResultDisplayed[0] = false;
            } else if ("=".equals(text)) {
                if (currentInput.length() > 0 && !operator[0].isEmpty()) {
                    double secondOperand = Double.parseDouble(currentInput.toString());
                    switch (operator[0]) {
```

```
                    case "+":
                        result[0] += secondOperand;
                        break;
                    case "-":
                        result[0] -= secondOperand;
                        break;
                    case "*":
                        result[0] *= secondOperand;
                        break;
                    case "/":
                        if (secondOperand != 0) {
                            result[0] /= secondOperand;
                        } else {
                            displayField.setText("Error");
                            currentInput.setLength(0);
                            operator[0] = "";
                            return;
                        }
                        break;
                }
                displayField.setText(String.valueOf(result[0]));
                currentInput.setLength(0);
                operator[0] = "";
                isResultDisplayed[0] = true;
            }
        }
    }
    });
}

// Make the frame visible
frame.setVisible(true);
}
}
```

OUTPUT


Advanced Calculator
9 +
7
8
9
/
4
5
6
*
1
2
3
-
0
.
=
+


Advanced Calculator
3
7
8
9
/
4
5
6
*
1
2
3
-
0
.
=
+


Advanced Calculator
12.0
7
8
9
/
4
5
6
*
1
2
3
-
0
.
=
+

QUES 7 - Develop a simple notepad application using Java Swing.

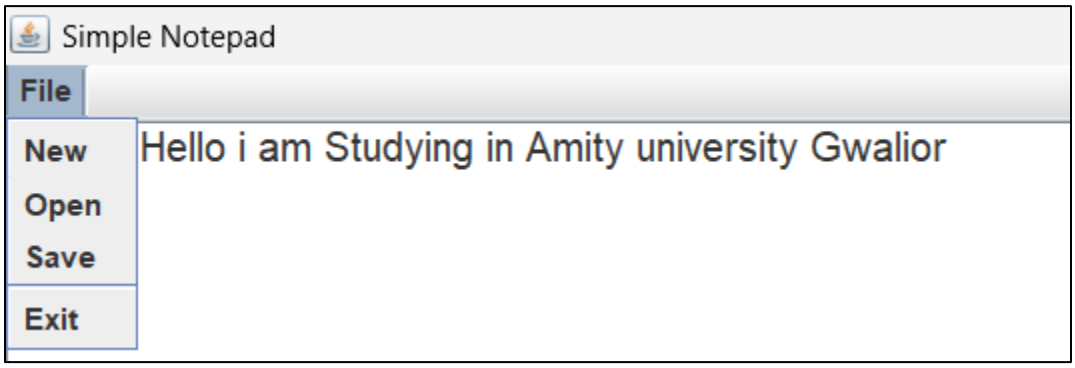
ANS

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

public class SimpleNotepad {
    public static void main(String[] args) {
        // Create the main frame
        JFrame frame = new JFrame("Simple Notepad");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(600, 400);

        // Create the text area
        JTextArea textArea = new JTextArea();
        textArea.setFont(new Font("Arial", Font.PLAIN, 16));
        JScrollPane scrollPane = new JScrollPane(textArea);
        frame.add(scrollPane, BorderLayout.CENTER);

        // Create the menu bar
        JMenuBar menuBar = new JMenuBar();

        // Create the File menu
        JMenu fileMenu = new JMenu("File");
        JMenuItem newItem = new JMenuItem("New");
        JMenuItem openItem = new JMenuItem("Open");
        JMenuItem saveItem = new JMenuItem("Save");
        JMenuItem exitItem = new JMenuItem("Exit");

        fileMenu.add(newItem);
        fileMenu.add(openItem);
        fileMenu.add(saveItem);
        fileMenu.addSeparator();
        fileMenu.add(exitItem);

        menuBar.add(fileMenu);
        frame.setJMenuBar(menuBar);

        // Action listeners for menu items
```

```
        // New file
        newItem.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                textArea.setText("");
            }
        });

        // Open file
        openItem.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JFileChooser fileChooser = new JFileChooser();
                int option = fileChooser.showOpenDialog(frame);
                if (option == JFileChooser.APPROVE_OPTION) {
                    File file = fileChooser.getSelectedFile();
                    try (BufferedReader reader = new BufferedReader(new FileReader(file))) {
                        textArea.setText("");
                        String line;
                        while ((line = reader.readLine()) != null) {
                            textArea.append(line + "\n");
                        }
                    } catch (IOException ex) {
                        JOptionPane.showMessageDialog(frame, "Error opening file.", "Error",
JOptionPane.ERROR_MESSAGE);
                    }
                }
            }
        });

        // Save file
        saveItem.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JFileChooser fileChooser = new JFileChooser();
                int option = fileChooser.showSaveDialog(frame);
                if (option == JFileChooser.APPROVE_OPTION) {
                    File file = fileChooser.getSelectedFile();
                    try (BufferedWriter writer = new BufferedWriter(new FileWriter(file))) {
                        writer.write(textArea.getText());
                    } catch (IOException ex) {
                        JOptionPane.showMessageDialog(frame, "Error saving file.", "Error",
JOptionPane.ERROR_MESSAGE);
```

```
                }
            }
        }
    });

    // Exit application
    exitItem.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            System.exit(0);
        }
    });

    // Make the frame visible
    frame.setVisible(true);
  }
}
```

OUTPUT –



QUES 8 - Demonstrate JDBC in a Java application by performing CRUD operations on a MySQL database. The application should include:

1. **Insertion of a record using a Statement.**
2. **Update of a record using a Statement.**
3. **Deletion of a record using a Statement.**
4. **Retrieval of records using a Statement.**
5. **Insertion of a record using a PreparedStatement with dynamic input from the user.**

ANS

```
package assignment1;

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

public class StudentCRUD {
    // Database credentials
    private static final String DB_URL = "jdbc:mysql://localhost:3306/ananya";
    private static final String USER = "root";
    private static final String PASSWORD = "xxx";

    public static void main(String[] args) {
        try (Connection conn = DriverManager.getConnection(DB_URL, USER, PASSWORD)) {
            createTable(conn);
            insertStudent(conn, 1, "Alice", 20, "Computer Science");
            insertStudent(conn, 2, "Bob", 21, "Mathematics");
            updateDepartment(conn, 1, "Physics");
            deleteStudent(conn, 2);
            retrieveAndDisplayAllStudents(conn);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    private static void createTable(Connection conn) throws SQLException {
        String createTableSQL = "CREATE TABLE IF NOT EXISTS Students ("
                + "StudentID INT PRIMARY KEY, "
                + "Name VARCHAR(50), "
                + "Age INT, "
                + "Department VARCHAR(50))";
        try (Statement stmt = conn.createStatement()) {
```

```
            stmt.execute(createTableSQL);
            System.out.println("Table Students created or already exists.");
        }
    }

    private static void insertStudent(Connection conn, int id, String name, int age, String
department) throws SQLException {
        String insertSQL = "INSERT INTO Students (StudentID, Name, Age, Department)
VALUES (?, ?, ?, ?)";
        try (PreparedStatement pstmt = conn.prepareStatement(insertSQL)) {
            pstmt.setInt(1, id);
            pstmt.setString(2, name);
            pstmt.setInt(3, age);
            pstmt.setString(4, department);
            pstmt.executeUpdate();
            System.out.println("Inserted student: " + name);
        }
    }

    private static void updateDepartment(Connection conn, int id, String newDepartment) throws
SQLException {
        String updateSQL = "UPDATE Students SET Department = ? WHERE StudentID = ?";
        try (PreparedStatement pstmt = conn.prepareStatement(updateSQL)) {
            pstmt.setString(1, newDepartment);
            pstmt.setInt(2, id);
            int rowsAffected = pstmt.executeUpdate();
            System.out.println("Updated department for StudentID " + id + ": " + (rowsAffected > 0
? "Success" : "Failed"));
        }
    }

    private static void deleteStudent(Connection conn, int id) throws SQLException {
        String deleteSQL = "DELETE FROM Students WHERE StudentID = ?";
        try (PreparedStatement pstmt = conn.prepareStatement(deleteSQL)) {
            pstmt.setInt(1, id);
            int rowsAffected = pstmt.executeUpdate();
            System.out.println("Deleted student with StudentID " + id + ": " + (rowsAffected > 0 ?
"Success" : "Failed"));
        }
    }

    private static void retrieveAndDisplayAllStudents(Connection conn) throws SQLException {
        String selectSQL = "SELECT * FROM Students";
        try (Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(selectSQL)) {
            System.out.println("Students Table:");
            while (rs.next()) {
```

```
            int studentID = rs.getInt("StudentID");
            String name = rs.getString("Name");
            int age = rs.getInt("Age");
            String department = rs.getString("Department");
            System.out.println("ID: " + studentID + ", Name: " + name + ", Age: " + age + ",
Department: " + department);
        }
    }
  }
}
}
```

Output:

```
Table Students created or already exists.
Inserted student: Anshika
Inserted student: Krishna
Inserted student: Mahi
Inserted student: Raha
Updated department for StudentID 4: Success
Deleted student with StudentID 3: Success
Students Table:
ID: 1, Name: Anshika, Age: 21, Department: Computer Science
ID: 2, Name: Krishna, Age: 20, Department: IT
ID: 4, Name: Raha, Age: 21, Department: Mathematics
```

QUES 9 Create a simple Servlet in Java that displays a "Hello World" message when accessed.

CODE -:

Hello.java

```
package com.amity;

import jakarta.servlet.ServletException;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;


public class Hello extends HttpServlet {
private static final long serialVersionUID = 1L;
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
PrintWriter out = response.getWriter();
out.println("<html>");
out.println("<body>");
out.println("<h1>Hello World</h1>");
out.println("</body>");
out.println("</html>");
}
}
```

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="https://jakarta.ee/xml/ns/jakartaee"
xsi:schemaLocation="https://jakarta.ee/xml/ns/jakartaee
https://jakarta.ee/xml/ns/jakartaee/web-app_6_0.xsd" id="WebApp_ID" version="6.0">

<display-name>File</display-name>
<welcome-file-list>
<welcome-file>index.html</welcome-file>
<welcome-file>index.jsp</welcome-file>
<welcome-file>index.htm</welcome-file>
<welcome-file>default.html</welcome-file>
<welcome-file>default.jsp</welcome-file>
<welcome-file>default.htm</welcome-file>
</welcome-file-list>
<!-- Servlet Configuration -->
```

```
<servlet>
<servlet-name>Hello</servlet-name>
<servlet-class>com.amity.Hello</servlet-class>
</servlet>
<!-- Servlet Mapping -->
<servlet-mapping>
<servlet-name>Hello</servlet-name>
<url-pattern>/hello</url-pattern>
</servlet-mapping>
</web-app>
```

Output:

QUES 10 - Create a web-based student login system using servlets and JSP.

ANS –

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<form action="loginServlet" method="post">
Student Name: <input type="text", name="sname"><br>
Password:<input type="password", name="spass"><br>
<input type=submit name=submit>
</form>
</body>
</html>
```

Loginfail.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Login Failed</title>
</head>
<body>
<h1>Login Failed</h1>
<p>Invalid username or password. Please try again.</p>
</body>
</html>
```

LoginServlet.java

```
package com.amity.servlet;

import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
```

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

public class LoginServlet extends HttpServlet {
private static final long serialVersionUID = 1L;

   public Connection con;
   @Override
public void init() throws ServletException{
          String dbUrl = getServletConfig().getInitParameter("DB_URL");
       String dbUser = getServletConfig().getInitParameter("DB_USER");
       String dbPass = getServletConfig().getInitParameter("DB_PASS");
try {
Class.forName("com.mysql.jdbc.Driver");
con = DriverManager.getConnection(dbUrl,dbUser,dbPass);
} catch (SQLException | ClassNotFoundException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
System.out.println(con);
   }

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
PrintWriter out = response.getWriter();
try {

String sname=request.getParameter("sname");
String spass= request.getParameter("spass");
String sql = "SELECT * FROM students WHERE sname = ? AND spass = ?";
       PreparedStatement st = con.prepareStatement(sql);
       st.setString(1, sname);
       st.setString(2, spass);

ResultSet rs = st.executeQuery();

//                    PrintWriter out = response.getWriter();
if (rs.next()) {
          request.setAttribute("username", sname);
          RequestDispatcher dispatcher = request.getRequestDispatcher("/WelcomeServlet");
          dispatcher.forward(request, response);  // Forward to WelcomeServlet on success
       } else {
          RequestDispatcher dispatcher = request.getRequestDispatcher("loginfail.html");
          dispatcher.include(request, response);  // Include loginfail.html on failure
       }
}catch(Exception e) {
```

```
out.println(e);
}
}
@Override
public void destroy() {
try {
con.close();
} catch (SQLException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
}

}
```

WelcomeServlet.java

```
package com.amity.servlet;

import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

@WebServlet("/WelcomeServlet")
public class WelcomeServlet extends HttpServlet {
private static final long serialVersionUID = 1L;

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
response.setContentType("text/html");
	PrintWriter out = response.getWriter();

	String username = (String) request.getAttribute("username");
	out.println("<html><body>");
	out.println("<h1>Welcome, " + username + "!</h1>");
	out.println("<p>Login successful. You are now in the Welcome Servlet.</p>");
	out.println("</body></html>");
}

}
```

Web.xml

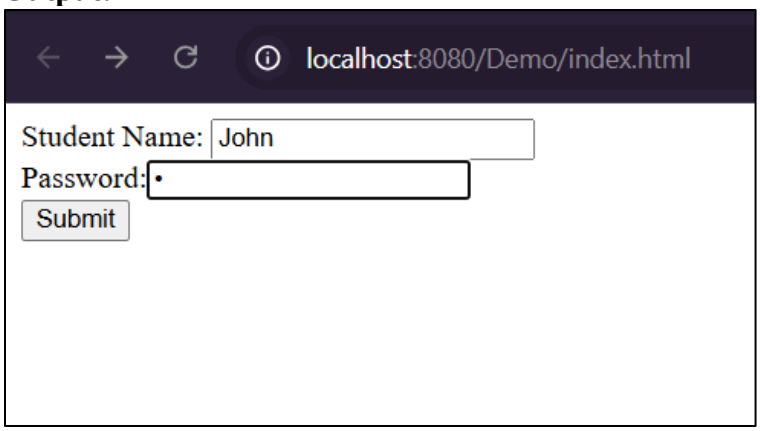
```
<?xml version="1.0" encoding="UTF-8"?>
```

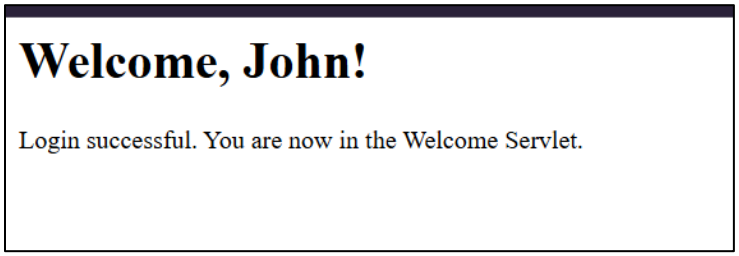
```
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="https://jakarta.ee/xml/ns/jakartaee"
xsi:schemaLocation="https://jakarta.ee/xml/ns/jakartaee
https://jakarta.ee/xml/ns/jakartaee/web-app_6_0.xsd" id="WebApp_ID" version="6.0">
<display-name>StudentManagement</display-name>
<welcome-file-list>
<welcome-file>index.html</welcome-file>
<welcome-file>index.jsp</welcome-file>
<welcome-file>index.htm</welcome-file>
<welcome-file>default.html</welcome-file>
<welcome-file>default.jsp</welcome-file>
<welcome-file>default.htm</welcome-file>
</welcome-file-list>
<servlet>
<display-name>LoginServlet</display-name>
<servlet-name>LoginServlet</servlet-name>
<servlet-class>com.amity.servlet.LoginServlet</servlet-class>
<init-param>
<param-name>DB_URL</param-name>
<param-value>jdbc:mysql://localhost:3307/mydb1</param-value>
</init-param>
<init-param>
<param-name>DB_USER</param-name>
<param-value>root</param-value>
</init-param>
<init-param>
<param-name>DB_PASS</param-name>
<param-value>abc123#</param-value>
</init-param>
</servlet>
<servlet-mapping>
<servlet-name>LoginServlet</servlet-name>
<url-pattern>/loginServlet</url-pattern>
</servlet-mapping>
<!--
<servlet>
<description></description>
<display-name>WelcomeServlet</display-name>
<servlet-name>WelcomeServlet</servlet-name>
<servlet-class>com.amity.servlet.WelcomeServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>WelcomeServlet</servlet-name>
<url-pattern>/WelcomeServlet</url-pattern>
</servlet-mapping>
-->
```

</web-app>

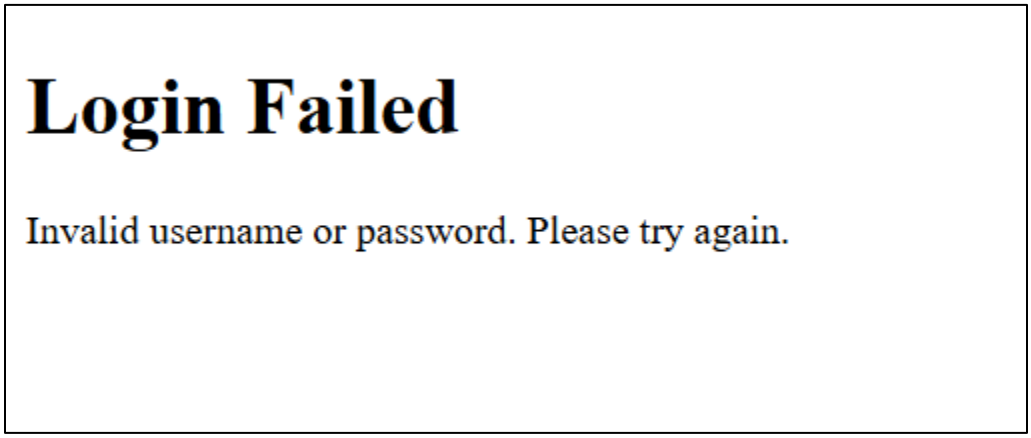
Sql Commands:
create database mydb1;
use mydb1;
insert into students()values('John','J');
SELECT * FROM students;

Output:





If user name or password is incorrect



QUES 11 - Create a simple web application to calculate the average of two numbers using the MVC architecture in servlets and JSP.

ANS –

Index.html

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
   <title>Average Of Two Numbers</title>
</head>
<body>
   <form action="averageController" method="post">
      First No: <input type="text" name="f1"><br>
      Second No: <input type="text" name="f2"><br>
      <input type="submit" value="Submit">
   </form>
</body>
</html>
```

AverageDisplay.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
   <title>Result Page</title>
</head>
<body>
   <% int result = (int) request.getAttribute("result"); %>
   <h1>Average of the two numbers is: <%= result %></h1>
</body>
</html>
```

AverageCalculator.java

```
package com.amity.servlet;
public class AverageCalculator {

	public int calculate(int x, int y) {
		return((x+y)/2);
	}
}
```

AverageController.java

```
package com.amity.servlet;
import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

@WebServlet ("/averageController")
public class AverageController extends HttpServlet {
	private static final long serialVersionUID = 1L;

	protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
//		PrintWriter out = response.getWriter();
//		out.println("hello");
		int x = Integer.parseInt(request.getParameter("f1"));
		int y = Integer.parseInt(request.getParameter("f2"));
		AverageCalculator model = new AverageCalculator();
		int z = model.calculate(x, y);
		request.setAttribute("result", z);
		RequestDispatcher rs =request.getRequestDispatcher("AverageDisplay.jsp");
		rs.forward(request, response);
	}

}
```

OUTPUT -:

First No: 15
Second No: 30
Submit

Average of the two numbers is: 22

QUES - 12 Demonstrate the use of Servlet Session Management.

ANS

Login.html

```
<!DOCTYPE html>
<html>
<head>
   <title>Login Page</title>
</head>
<body>
   <h2>Login</h2>
   <form action="LoginServlet" method="post">
      Username: <input type="text" name="username"><br>
      <input type="submit" value="Login">
   </form>
</body>
</html>
```

LoginServlet.java

```
package com.amity;

import java.io.IOException;
import jakarta.servlet.ServletException;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import jakarta.servlet.http.HttpSession;

public class LoginServlet extends HttpServlet {
   protected void doPost(HttpServletRequest request, HttpServletResponse response)
         throws ServletException, IOException {

      String username = request.getParameter("username");
            if (username != null && !username.isEmpty()) {

         HttpSession session = request.getSession();
         session.setAttribute("user", username);

         response.sendRedirect("ProfileServlet");
      } else {
         response.getWriter().println("Invalid Username! Please go back and enter again.");
      }
   } }
```

ProfileServlet.java

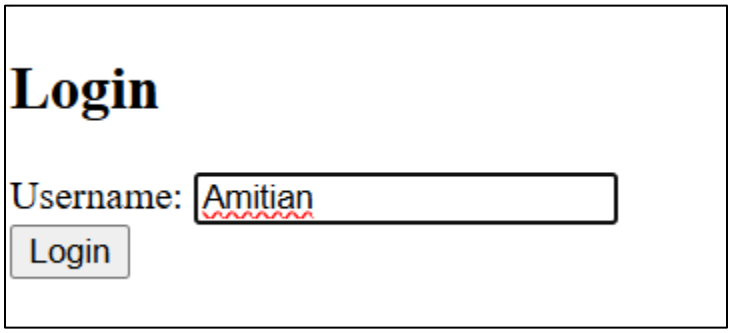
```
package com.amity;

import java.io.IOException;
import jakarta.servlet.ServletException;

import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import jakarta.servlet.http.HttpSession;


public class ProfileServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        HttpSession session = request.getSession(false);

        response.setContentType("text/html");
        if (session != null && session.getAttribute("user") != null) {
            String username = (String) session.getAttribute("user");
            response.getWriter().println("<h2>Welcome, " + username + "</h2>");
            response.getWriter().println("<a href='LogoutServlet'>Logout</a>");
        } else {
            response.getWriter().println("<h2>No active session. Please <a
href='login.html'>login</a>.</h2>");
        }
    }
}
```

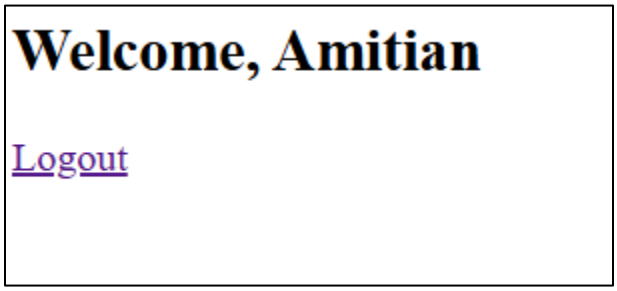
LogoutServlet.java

```
package com.amity;

import java.io.IOException;
import jakarta.servlet.ServletException;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import jakarta.servlet.http.HttpSession;

public class LogoutServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
```

```
        HttpSession session = request.getSession(false);

        if (session != null) {
            session.invalidate(); // Invalidate the session
        }

        response.getWriter().println("<h2>You have been logged out successfully!</h2>");
        response.getWriter().println("<a href='login.html'>Login Again</a>");
    }
}
```

OUTPUT-:






```
<h2>You have been logged out successfully!</h2>
<a href='login.html'>Login Again</a>
```

QUES – 13 Program of EJB

ANS

OperationSessionBean.java

```
package com.javacodegeeks.example.ejb;
import javax.ejb.Stateless;
public class OperationsSessionBean implements OperationsSessionBeanRemote {
    public float add(float x, float y) {
        return x + y;
    }
    public float subtract(float x, float y) {
        return x - y;
    }
    public float mutliply(float x, float y) {
        return x * y;
    }
    public float divide(float x, float y) {
        return x / y;
    }
}
```

Form.jsp

```
<html>
    <head>
        <title>Calculator</title>
    </head>
    <body bgcolor="lightgreen">
        <h1>Basic Operations</h1>
        <hr>
        <form action="Result.jsp" method="POST">
```

```
            <p>Enter first value:
                <input type="text" name="num1" size="25"></p>
            <br>
            <p>Enter second value:
                <input type="text" name="num2" size="25"></p>
            <br>
            <b>Select your choice:</b><br>
            <input type="radio" name="group1" value ="add">Addition<br>
            <input type="radio" name="group1" value ="sub">Subtraction<br>
            <input type="radio" name="group1" value ="multi">Multiplication<br>
            <input type="radio" name="group1" value ="div">Division<br>
            <p>
                <input type="submit" value="Submit">
                <input type="reset" value="Reset">
           </p>
        </form>
    </body>
</html>
</form>
```

Result.jsp

```
<%@ page contentType="text/html; charset=UTF-8" %>
<%@ page import="com.javacodegeeks.example.ejb.*, javax.naming.*"%>
<%!
    private OperationsSessionBeanRemote ops = null;
    float result = 0;
    public void jspInit() {
        try {
            InitialContext ic = new InitialContext();
            ops =
```

```
(OperationsSessionBeanRemote)ic.lookup(OperationsSessionBeanRemote.class.getName());
            System.out.println("Loaded Calculator Bean");
        } catch (Exception ex) {
            System.out.println("Error:"
                    + ex.getMessage());
        }
    }
    public void jspDestroy() {
        ops = null;
    }
%>
<%
    try {
        String s1 = request.getParameter("num1");
        String s2 = request.getParameter("num2");
        String s3 = request.getParameter("group1");
        System.out.println(s3);
        if (s1 != null && s2 != null) {
            Float num1 = new Float(s1);
            Float num2 = new Float(s2);
            if (s3.equals("add")) {
                result = ops.add(num1.floatValue(), num2.floatValue());
            } else if (s3.equals("sub")) {
                result = ops.subtract(num1.floatValue(), num2.floatValue());
            } else if (s3.equals("multi")) {
                result = ops.mutliply(num1.floatValue(), num2.floatValue());
            } else {
                result = ops.divide(num1.floatValue(), num2.floatValue());
            }
%>
<p>
```

```
    <b>The result is:</b> <%= result%>
<p>
    <%
            }
        }// end of try
        catch (Exception e) {
            e.printStackTrace();
            //result = "Not valid";
        }
    %>
```

OUTPUT-: