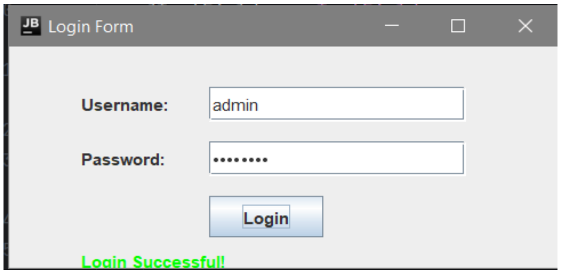
Experiment – 9

1. Write a Java Swing program to create a Login Form using JTextField, JPasswordField, JButton, and JLabel. When the login button is clicked, validate if the username is "admin" and the password is "password".

   Code :-

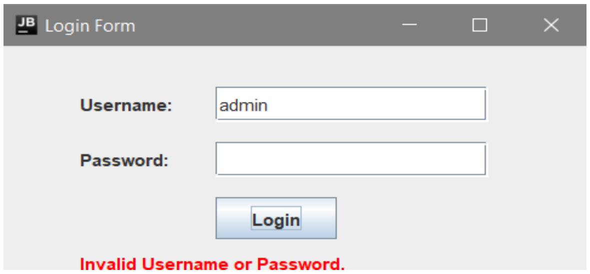
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
public class LoginForm extends JFrame implements ActionListener {
    // Components
    JLabel userLabel, passLabel, messageLabel;
    JTextField userTextField;
    JPasswordField passField;
    JButton loginButton;
    public LoginForm() {
        // Frame setup
        setTitle("Login Form");
        setSize(400, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(null); // Using absolute positioning
        setLocationRelativeTo(null); // Center the window
        // --- Username ---
        userLabel = new JLabel("Username:");
        userLabel.setBounds(50, 30, 80, 25);
        add(userLabel);
        userTextField = new JTextField();
        userTextField.setBounds(140, 30, 180, 25);
        add(userTextField);
        // --- Password ---
        passLabel = new JLabel("Password:");
        passLabel.setBounds(50, 70, 80, 25);
        add(passLabel);

        passField = new JPasswordField();
        passField.setBounds(140, 70, 180, 25);
        add(passField);

        // --- Login Button ---
        loginButton = new JButton("Login");
        loginButton.setBounds(140, 110, 80, 30);
        loginButton.addActionListener(this); // Register listener
        add(loginButton);

        // --- Message Label (for feedback) ---
        messageLabel = new JLabel("");
        messageLabel.setBounds(50, 145, 300, 25);
        messageLabel.setForeground(Color.RED);
        add(messageLabel);

        setVisible(true);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == loginButton) {
            String username = userTextField.getText();
            String password = new String(passField.getPassword()); // Get password as String

            // Validation
            if (username.equals("admin") && password.equals("password")) {
                messageLabel.setForeground(Color.GREEN);
                messageLabel.setText("Login Successful!");
                // JOptionPane.showMessageDialog(this, "Login Successful!");
                // dispose(); // Close the login window
            } else {
                messageLabel.setForeground(Color.RED);
                messageLabel.setText("Invalid Username or Password.");

            }
        }
    }
    public static void main(String[] args) {
        // Run the GUI code on the Event Dispatch Thread (EDT)
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                new LoginForm();
            }
        });
    }
}
```



Username: admin

Password: ••••••••

Login Successful!



Username: admin

Password:

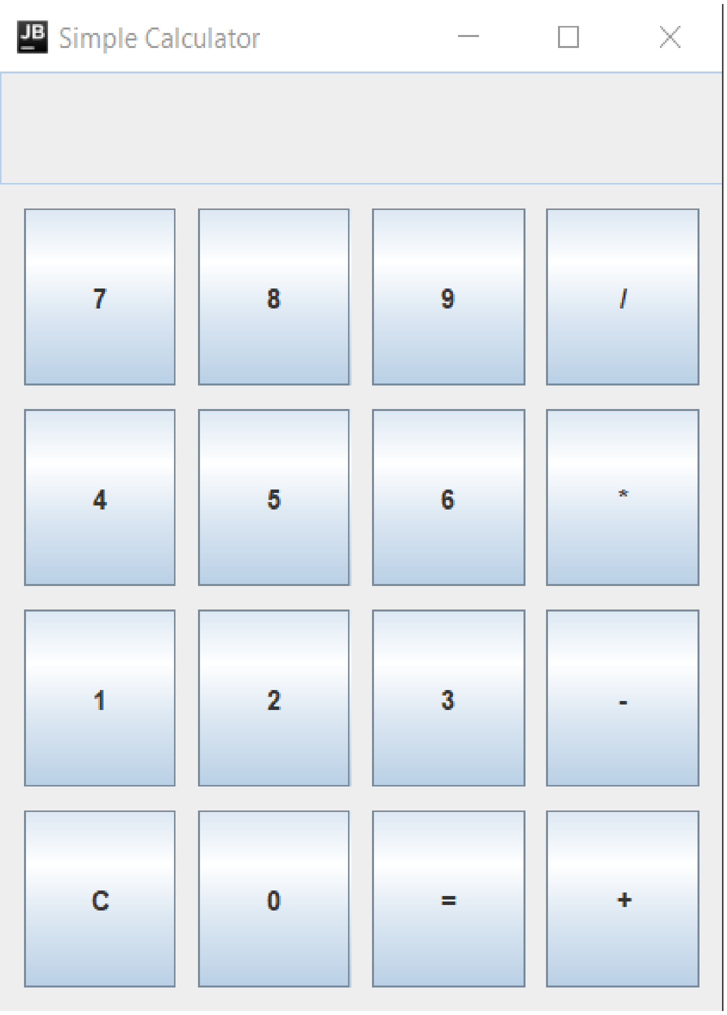
Invalid Username or Password.

2. Design a simple calculator using Java Swing with buttons for digits (0-9), addition (+), subtraction (-), multiplication (*), and division (/). Implement event handling for button clicks and display the result in a JTextField.

Code :-

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

public class SimpleCalculator extends JFrame implements ActionListener {

    // Components
    JTextField displayField;
    JButton[] numberButtons = new JButton[10]; // 0-9
    JButton addButton, subButton, mulButton, divButton;
    JButton equButton, clrButton;
    JPanel buttonPanel;

    // Calculation variables
    double num1 = 0, num2 = 0, result = 0;
    char operator;
    boolean newCalculation = true; // Flag to clear display for new number

    public SimpleCalculator() {
        // Frame setup
        setTitle("Simple Calculator");
        setSize(350, 450);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout()); // Use BorderLayout
        setLocationRelativeTo(null);

        // --- Display Field ---
        displayField = new JTextField();
        displayField.setPreferredSize(new Dimension(300, 50));
        displayField.setFont(new Font("Arial", Font.PLAIN, 24));
        displayField.setHorizontalAlignment(JTextField.RIGHT);
        displayField.setEditable(false); // User shouldn't type directly
        add(displayField, BorderLayout.NORTH);

        // --- Button Panel ---
        buttonPanel = new JPanel();
        buttonPanel.setLayout(new GridLayout(4, 4, 10, 10)); // 4x4 grid with gaps
buttonPanel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10)); // Padding

// Number Buttons (7, 8, 9, /)
for (int i = 7; i <= 9; i++) {
    numberButtons[i] = new JButton(String.valueOf(i));
    numberButtons[i].addActionListener(this);
    buttonPanel.add(numberButtons[i]);
}
divButton = new JButton("/");
divButton.addActionListener(this);
buttonPanel.add(divButton);

// Number Buttons (4, 5, 6, *)
for (int i = 4; i <= 6; i++) {
    numberButtons[i] = new JButton(String.valueOf(i));
    numberButtons[i].addActionListener(this);
    buttonPanel.add(numberButtons[i]);
}
mulButton = new JButton("*");
mulButton.addActionListener(this);
buttonPanel.add(mulButton);

// Number Buttons (1, 2, 3, -)
for (int i = 1; i <= 3; i++) {
    numberButtons[i] = new JButton(String.valueOf(i));
    numberButtons[i].addActionListener(this);
    buttonPanel.add(numberButtons[i]);
}
subButton = new JButton("-");
subButton.addActionListener(this);
buttonPanel.add(subButton);

// Bottom Row (C, 0, =, +)
clrButton = new JButton("C");
clrButton.addActionListener(this);
buttonPanel.add(clrButton);

numberButtons[0] = new JButton("0");
numberButtons[0].addActionListener(this);
buttonPanel.add(numberButtons[0]);

equButton = new JButton("=");
equButton.addActionListener(this);
buttonPanel.add(equButton);

addButton = new JButton("+");
addButton.addActionListener(this);
buttonPanel.add(addButton);
```

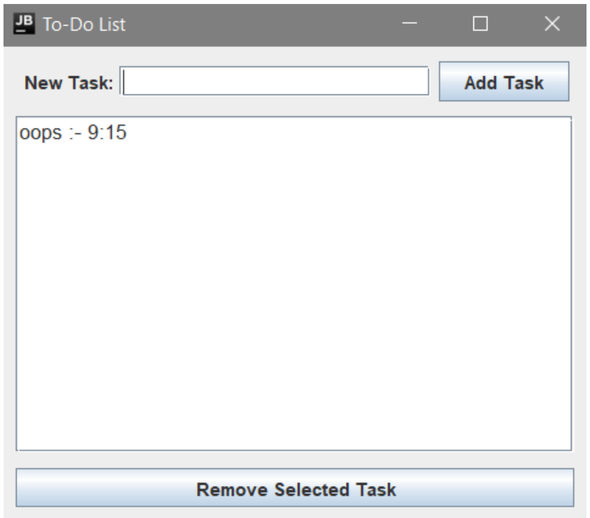
```
    add(buttonPanel, BorderLayout.CENTER);
    setVisible(true);
}

@Override
public void actionPerformed(ActionEvent e) {
    String command = e.getActionCommand();
    // Handle Number Clicks
    for (int i = 0; i < 10; i++) {
        if (command.equals(String.valueOf(i))) {
            if (newCalculation) {
                displayField.setText(""); // Clear if starting new number
                newCalculation = false;
            }
            displayField.setText(displayField.getText() + command);
            return; // Exit after handling number
        }
    }
    // Handle Operator Clicks
    switch (command) {
        case "+", "-", "*", "/" -> {
            if (!displayField.getText().isEmpty()) {
                num1 = Double.parseDouble(displayField.getText());
                operator = command.charAt(0);
                newCalculation = true; // Expecting second number next
            }
        }
        // Handle Equals Click
        case "=" -> {
            if (!displayField.getText().isEmpty() && !newCalculation) {
                num2 = Double.parseDouble(displayField.getText());
                try {
                    switch (operator) {
                        case '+' -> result = num1 + num2;
                        case '-' -> result = num1 - num2;
                        case '*' -> result = num1 * num2;
                        case '/' -> {
                            if (num2 == 0) {
                                throw new ArithmeticException("Division by zero");
                            }
                            result = num1 / num2;
                        }
                    }
                    displayField.setText(String.valueOf(result));
                    num1 = result; // Allow chaining calculations
                } catch (ArithmeticException ex) {
                    displayField.setText("Error: Div by 0");
                    } catch (Exception ex) {
                        displayField.setText("Error");
                    }
                    newCalculation = true; // Start fresh after equals
                }
            }
            // Handle Clear Click
            case "C" -> {
                displayField.setText("");
                num1 = num2 = result = 0;
                operator = '\0'; // Reset operator
                newCalculation = true;
            }
        }
    }

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



3. Write a Java Swing program to implement a To-Do List using JList. Provide: a. A JTextField to enter tasks
   b. An Add button to add tasks to the list
   c. A Remove button to delete selected tasks

Code :-

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class TodoListApp extends JFrame implements ActionListener {
    // Components
    JTextField taskInputField;
    JButton addButton, removeButton;
    JList<String> taskList;
    DefaultListModel<String> listModel; // Model to manage list data
    JScrollPane listScrollPane; // To make the list scrollable

    public TodoListApp() {
        // Frame setup
        setTitle("To-Do List");
        setSize(400, 350);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout(10, 10)); // BorderLayout with gaps
        setLocationRelativeTo(null);
        getRootPane().setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10)); // Padding for the whole frame

        // --- Input Panel (Top) ---
        JPanel inputPanel = new JPanel(new FlowLayout(FlowLayout.LEFT, 5, 0)); // FlowLayout for TextField and Add button
        taskInputField = new JTextField(20); // Set preferred width
        addButton = new JButton("Add Task");
        addButton.addActionListener(this);

        inputPanel.add(new JLabel("New Task:"));
        inputPanel.add(taskInputField);
        inputPanel.add(addButton);
        add(inputPanel, BorderLayout.NORTH);

        // --- List (Center) ---
        listModel = new DefaultListModel<>();
        taskList = new JList<>(listModel);
        taskList.setSelectionMode(ListSelectionModel.SINGLE_SELECTION); // Allow only one item
selection
        taskList.setFont(new Font("Arial", Font.PLAIN, 14));
        listScrollPane = new JScrollPane(taskList); // Add list to scroll pane
        add(listScrollPane, BorderLayout.CENTER);

        // --- Remove Button (Bottom) ---
        removeButton = new JButton("Remove Selected Task");
        removeButton.addActionListener(this);
        add(removeButton, BorderLayout.SOUTH);

        setVisible(true);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == addButton) {
            // Add Task
            String task = taskInputField.getText().trim(); // Get text and remove leading/trailing spaces
            if (!task.isEmpty()) {
                listModel.addElement(task); // Add to the list model
                taskInputField.setText(""); // Clear the input field
                taskInputField.requestFocusInWindow(); // Set focus back to input field
            } else {
                JOptionPane.showMessageDialog(this, "Please enter a task.", "Input Error",
JOptionPane.WARNING_MESSAGE);
            }
        } else if (e.getSource() == removeButton) {
            // Remove Task
            int selectedIndex = taskList.getSelectedIndex();
            if (selectedIndex != -1) { // Check if an item is actually selected
                listModel.removeElementAt(selectedIndex); // Remove from the model
            } else {
                JOptionPane.showMessageDialog(this, "Please select a task to remove.", "Selection Error",
JOptionPane.WARNING_MESSAGE);
            }
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(TodoListApp::new);
    }
}
```



oops :- 9:15

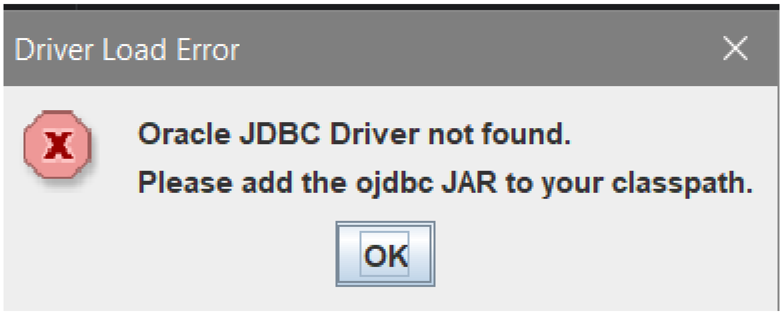
4. Write a Swing application with JDBC to insert and retrieve employee details (ID, Name, Department, Salary) into/from a MySQL database. Use d. JTextField for input e. JButton for inserting data f. JTable for displaying retrieved records

Code :-

```
 import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.*;
import java.sql.*; // Import necessary SQL classes

// Rename class to match filename if desired, or keep as EmployeeDBApp
public class EmployeeDBApp_Oracle extends JFrame implements ActionListener {

    // --- Oracle Database Configuration ---
    // !! IMPORTANT: Replace with your actual Oracle connection details !!
    // Format: jdbc:oracle:thin:@<hostname>:<port>:<service_name_or_sid>
    // Example: jdbc:oracle:thin:@your_oracle_host.com:1521:yourservicename
    private static final String DB_URL = "jdbc:oracle:thin:@your_hostname:1521:your_service_name"; //
<<< CHANGE THIS >>>
    private static final String DB_USER = "your_oracle_username"; // <<< CHANGE THIS (Often APEX
Workspace Name) >>>
    private static final String DB_PASSWORD = "your_oracle_password"; // <<< CHANGE THIS >>>
    private static final String ORACLE_DRIVER = "oracle.jdbc.driver.OracleDriver"; // Standard Oracle
driver class

    // --- GUI Components ---
    JTextField idField, nameField, deptField, salaryField;
    JButton insertButton, refreshButton;
    JTable employeeTable;
    DefaultTableModel tableModel;
    JScrollPane tableScrollPane;

    public EmployeeDBApp_Oracle() { // Constructor matches class name
        // Frame setup
        setTitle("Employee Oracle DB Interaction"); // Updated title
        setSize(600, 450);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout(10, 10));
        setLocationRelativeTo(null);
        getRootPane().setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
        // --- Input Panel (Top) ---
        JPanel inputPanel = new JPanel(new GridLayout(5, 2, 5, 5));

        inputPanel.add(new JLabel("Employee ID:"));
        idField = new JTextField();
        inputPanel.add(idField);

        inputPanel.add(new JLabel("Name:"));
        nameField = new JTextField();
        inputPanel.add(nameField);

        inputPanel.add(new JLabel("Department:"));
        deptField = new JTextField();
        inputPanel.add(deptField);

        inputPanel.add(new JLabel("Salary:"));
        salaryField = new JTextField();
        inputPanel.add(salaryField);

        insertButton = new JButton("Insert Employee");
        insertButton.addActionListener(this);
        inputPanel.add(insertButton);

        refreshButton = new JButton("Refresh Table");
        refreshButton.addActionListener(this);
        inputPanel.add(refreshButton);

        add(inputPanel, BorderLayout.NORTH);

        // --- Table (Center) ---
        String[] columnNames = {"ID", "Name", "Department", "Salary"};
        tableModel = new DefaultTableModel(columnNames, 0) {
            @Override
            public boolean isCellEditable(int row, int column) {
                return false;
            }
        };
        employeeTable = new JTable(tableModel);
        employeeTable.setFillsViewportHeight(true);
        tableScrollPane = new JScrollPane(employeeTable);
        add(tableScrollPane, BorderLayout.CENTER);

        // Load initial data
        loadTableData();

        setVisible(true);
    }
```

```
// --- Action Listener ---
@Override
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == insertButton) {
        insertEmployeeData();
    } else if (e.getSource() == refreshButton) {
        loadTableData();
    }
}

// --- Database Methods ---

// Method to load data from Oracle DB into the JTable
private void loadTableData() {
    tableModel.setRowCount(0); // Clear existing data

    // SQL query is standard and should work on Oracle
    String sql = "SELECT ID, Name, Department, Salary FROM employees ORDER BY ID";

    // Use try-with-resources for automatic resource closing
    try (Connection conn = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD);
         Statement stmt = conn.createStatement();
         ResultSet rs = stmt.executeQuery(sql)) {

        while (rs.next()) {
            // Retrieve data by column name
            int id = rs.getInt("ID");
            String name = rs.getString("Name");
            String dept = rs.getString("Department");
            double salary = rs.getDouble("Salary");

            // Add row to the table model
            tableModel.addRow(new Object[]{id, name, dept, salary});
        }

    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(this, "Error loading data from Oracle: " + ex.getMessage(),
                "Database Error", JOptionPane.ERROR_MESSAGE);
        ex.printStackTrace();
    }
}

// Method to insert data from text fields into the Oracle DB
private void insertEmployeeData() {
    String idStr = idField.getText().trim();
    String name = nameField.getText().trim();
    String dept = deptField.getText().trim();
    String salaryStr = salaryField.getText().trim();
```

```
        if (idStr.isEmpty() || name.isEmpty() || dept.isEmpty() || salaryStr.isEmpty()) {
            JOptionPane.showMessageDialog(this, "Please fill in all fields.", "Input Error",
JOptionPane.WARNING_MESSAGE);
            return;
        }

        int id;
        double salary;
        try {
            id = Integer.parseInt(idStr);
            salary = Double.parseDouble(salaryStr);
        } catch (NumberFormatException ex) {
            JOptionPane.showMessageDialog(this, "Invalid ID or Salary format (Numbers required).", "Input
Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        // SQL INSERT statement (standard, should work)
        // Using PreparedStatement for security and efficiency
        String sql = "INSERT INTO employees (ID, Name, Department, Salary) VALUES (?, ?, ?, ?)";

        try (Connection conn = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD);
             PreparedStatement pstmt = conn.prepareStatement(sql)) {

            pstmt.setInt(1, id);
            pstmt.setString(2, name);
            pstmt.setString(3, dept);
            pstmt.setDouble(4, salary);

            int rowsAffected = pstmt.executeUpdate();

            if (rowsAffected > 0) {
                JOptionPane.showMessageDialog(this, "Employee inserted successfully!", "Success",
JOptionPane.INFORMATION_MESSAGE);
                idField.setText("");
                nameField.setText("");
                deptField.setText("");
                salaryField.setText("");
                loadTableData(); // Refresh table
            } else {
                JOptionPane.showMessageDialog(this, "Failed to insert employee (0 rows affected).", "Insert
Failed", JOptionPane.WARNING_MESSAGE);
            }

        } catch (SQLIntegrityConstraintViolationException ex) {
            // This specific exception should catch Primary Key violations in Oracle too
            JOptionPane.showMessageDialog(this, "Error: Employee ID already exists (Primary Key
Violation).", "Database Error", JOptionPane.ERROR_MESSAGE);
      ex.printStackTrace();
    } catch (SQLException ex) {
      // Catch other potential Oracle errors (e.g., connection issues, syntax errors)
      JOptionPane.showMessageDialog(this, "Database error during insertion: " + ex.getMessage(),
"Database Error", JOptionPane.ERROR_MESSAGE);
      ex.printStackTrace();
    }
  }

  // --- Main Method ---
  public static void main(String[] args) {
    // Explicitly load the Oracle JDBC driver
    try {
      Class.forName(ORACLE_DRIVER);
    } catch (ClassNotFoundException e) {
      System.err.println("FATAL ERROR: Oracle JDBC Driver not found.");
      System.err.println("Please ensure the Oracle JDBC JAR (e.g., ojdbc8.jar) is in your classpath.");
      e.printStackTrace();
      // Display a graphical error message as well
      JOptionPane.showMessageDialog(null,
          "Oracle JDBC Driver not found.\nPlease add the ojdbc JAR to your classpath.",
          "Driver Load Error", JOptionPane.ERROR_MESSAGE);
      return; // Exit if driver not found
    }

    // Run the GUI on the Event Dispatch Thread
    // Ensure you instantiate the correct class name here
    SwingUtilities.invokeLater(EmployeeDBApp_Oracle::new);
  }
}
```



Driver Load Error

Oracle JDBC Driver not found.
Please add the ojdbc JAR to your classpath.

Note :- my apex schema is not working properly therefore I cannot create a table.