```java
// Slip - 1 Q1 [JAVA]
public class S1Q1 extends Thread {
    public void run() {
        try {
            for(char c = 'A'; c <= 'Z'; c++) {
                System.out.println("Generated Charachter: " + c);
                Thread.sleep(2000);
            }
        } catch(InterruptedException e) {
            System.out.println(e);
        }
    }

    public static void main(String args[]) {
        S1Q1 t1 = new S1Q1();
        t1.start();
    }
}

// Slip - 1 Q2 [JAVA]
import java.sql.*;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import javax.swing.table.DefaultTableModel;

public class S1Q2 extends JFrame implements ActionListener {
    JLabel enoLabel, enameLabel, designationLabel, salaryLabel;
    JTextField enoField, enameField, designationField, salaryField;
    JButton saveBtn, displayBtn;
    JPanel panel, displayPanel;

    public S1Q2() {
        setTitle("Employee Detail Form");
        panel = new JPanel(new GridLayout(5,2));
        displayPanel = new JPanel(new BorderLayout());

        enoLabel = new JLabel("Employee Number:");
        panel.add(enoLabel);

        enoField = new JTextField(20);
        panel.add(enoField);

        enameLabel = new JLabel("Employee Name:");
        panel.add(enameLabel);

        enameField = new JTextField(20);
        panel.add(enameField);
        designationLabel = new JLabel("Employee Designation:");
        panel.add(designationLabel);

        designationField = new JTextField(20);
        panel.add(designationField);

        salaryLabel = new JLabel("Employee Salary:");
        panel.add(salaryLabel);

        salaryField = new JTextField(20);
        panel.add(salaryField);

        saveBtn = new JButton("Save Details");
        saveBtn.addActionListener(this);
        panel.add(saveBtn);

        displayBtn = new JButton("Display Details");
        displayBtn.addActionListener(this);
        panel.add(displayBtn);

        add(panel, BorderLayout.PAGE_START);
        add(displayPanel, BorderLayout.CENTER);

        setSize(450,300);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public void actionPerformed(ActionEvent e) {
        if(e.getSource() == saveBtn)
            saveEmployeeDetails();
        else if(e.getSource() == displayBtn)
            displayEmployeeDetails();
    }
    public void saveEmployeeDetails() {
        String strEno = enoField.getText();
        int eno = Integer.parseInt(strEno);
        String ename = enameField.getText();
        String designation = designationField.getText();
        String salary = salaryField.getText();

        try {
            Class.forName("org.postgresql.Driver");
            Connection con
DriverManager.getConnection("jdbc:postgresql://localhost:5432/ty92", "ty92",
"ty92");
```

```
PreparedStatement ps = con.prepareStatement("INSERT INTO Employee (ENO, ENAME,
DESIGNATION, SALARY) VALUES (?, ?, ?, ?)");
            ps.setInt(1, eno);
            ps.setString(2, ename);
            ps.setString(3, designation);
            ps.setString(4, salary);

            int rowsAffected = ps.executeUpdate();

            if (rowsAffected > 0) {
                JOptionPane.showMessageDialog(this, "Employee details saved
successfully");
            } else {
                JOptionPane.showMessageDialog(this, "Failed to save employee
details", "Error", JOptionPane.ERROR_MESSAGE);
            }

            ps.close();
            con.close();
        } catch (ClassNotFoundException | SQLException e) {
            System.out.println(e);
        }
    }
public void displayEmployeeDetails() {
try {
     Class.forName("org.postgresql.Driver");
      Connection con =
DriverManager.getConnection("jdbc:postgresql://localhost:5432/ty92", "ty92",
"ty92");
            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery("SELECT * FROM Employee");

            DefaultTableModel model = new DefaultTableModel();
            model.addColumn("Employee Number");
            model.addColumn("Employee Name");
            model.addColumn("Employee Designation");
            model.addColumn("Employee Salary");

            while (rs.next()) {
                Object[] row = {
                    rs.getInt("ENO"),
                    rs.getString("ENAME"),
                    rs.getString("DESIGNATION"),
                    rs.getString("SALARY")
                };
                model.addRow(row);
            }
            JTable table = new JTable(model);
            JScrollPane scrollPane = new JScrollPane(table);

            displayPanel.removeAll();
            displayPanel.add(scrollPane);
            displayPanel.revalidate();
            displayPanel.repaint();

            rs.close();
            stmt.close();
            con.close();
        } catch (ClassNotFoundException | SQLException e) {
            System.out.println(e);
        }
    }

    public static void main(String args[]) {
        new S1Q2();
    }
}
// Slip - 2 Q1 [JAVA]
import java.util.*;
public class S2Q1 {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);

        System.out.print("\nEnter total number of friends: ");
        int n = sc.nextInt();
        sc.nextLine();
        HashSet<String> friendsSet = new HashSet<String>();
        for(int i=0; i<n; i++) {
            System.out.print("\nEnter the name of friend " + (i+1) + ": ");
            String name = sc.nextLine();
            friendsSet.add(name);
        }
ArrayList<String> sortedFriendsList = new ArrayList<String>(friendsSet);
        Collections.sort(sortedFriendsList);

        System.out.println("\n- Sorted friend's list -\n");

        for(String friend : sortedFriendsList)
            System.out.println(friend);

        sc.close();
    }
}
// Slip - 2 Q2 [JAVA]
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class S2Q2 extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        // Set response content type
        response.setContentType("text/html");

        // Get server information
        String serverInfo = getServletContext().getServerInfo();

        // Get servlet names
        Collection<? extends ServletRegistration> servletRegistrations =
getServletContext().getServletRegistrations().values();

        // Output HTML response
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head><title>Servlet Information</title></head>");
        out.println("<body>");
        out.println("<h2>Server Information:</h2>");
        out.println("<p>Server Software: " + serverInfo + "</p>");
        out.println("<h2>Loaded Servlets:</h2>");
        for (ServletRegistration servletRegistration : servletRegistrations) {
            out.println("<p>" + servletRegistration.getName() + "</p>");
        }
        out.println("</body>");
        out.println("</html>");
    }
}

<!-- Slip - 3 Q1 [JAVA] -->
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Patient Details</title>
</head>
<body>

<h2>Patient Details</h2>

<table border="1">
  <tr>
    <th>Patient Number</th>
```

```
        <th>Patient Name</th>
        <th>Address</th>
        <th>Age</th>
        <th>Disease</th>
      </tr>
      <%
        // Sample patient data (replace this with actual patient data)
        String[][] patients = {
            {"P001", "John Doe", "123 Main St", "35", "Fever"},
            {"P002", "Jane Smith", "456 Elm St", "42", "Headache"},
            {"P003", "David Johnson", "789 Oak St", "28", "Allergy"}
        };

        // Loop through each patient and display their details in the table
        for (String[] patient : patients) {
      %>
      <tr>
        <td><%= patient[0] %></td>
        <td><%= patient[1] %></td>
        <td><%= patient[2] %></td>
        <td><%= patient[3] %></td>
        <td><%= patient[4] %></td>
      </tr>
      <%
        }
      %>
    </table>

    </body>
    </html>

    // Slip - 3 Q2 [JAVA]
    import java.util.*;
    public class S3Q2 {
        public static void main(String args[]) {
            LinkedList<String> l1 = new LinkedList<String>();

            l1.add("Apple");
            l1.add("Banana");
            l1.add("Orange");

            System.out.println("\nOriginal linked list: ");
            System.out.println(l1);
          l1.removeFirst();
          System.out.println("\nLinked list after removing the first element: ");
            System.out.println(l1);
            System.out.println("\n- Linked list in reverse order - \n");
          ListIterator<String> itr = l1.listIterator(l1.size());
```

```
 while(itr.hasPrevious())
            System.out.println(itr.previous());
    }
}
// Slip - 4 Q1 [JAVA]
import java.awt.*;
import javax.swing.*;
import java.util.concurrent.TimeUnit;

public class S4Q1 implements Runnable {

    private JLabel label;

    public S4Q1(JLabel label) {
        this.label = label;
    }

    public void run() {
        try {
            while (true) {
                label.setVisible(true);
        TimeUnit.MILLISECONDS.sleep(500); // Text visible for 500 milliseconds
                label.setVisible(false);
      TimeUnit.MILLISECONDS.sleep(500); // Text invisible for 500 milliseconds
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
    public static void main(String[] args) {
        JFrame frame = new JFrame("Blinking Text");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(300, 200);
        JLabel label = new JLabel("Blinking Text");
        label.setFont(new Font("Arial", Font.PLAIN, 20));
        label.setHorizontalAlignment(JLabel.CENTER);
        frame.getContentPane().add(label, BorderLayout.CENTER);
        S4Q1 blinkingText = new S4Q1(label);
        Thread thread = new Thread(blinkingText);
        thread.start();
        frame.setVisible(true);
    }
}
// Slip - 4 Q2 [JAVA]
import java.awt.*;
import java.util.*;
import javax.swing.*;
import java.awt.event.*;
```

```java
public class S4Q2 extends JFrame {

    private Map<String, String> cityCodes;

    private JTextField cityNameTextField;
    private JTextField stdCodeTextField;

    public S4Q2() {
        cityCodes = new HashMap<>();

        setTitle("City Code App");
        setSize(300, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        initUI();
    }

    private void initUI() {
        JPanel panel = new JPanel();
        panel.setLayout(new FlowLayout());

        cityNameTextField = new JTextField(20);
        stdCodeTextField = new JTextField(20);

        JButton addButton = new JButton("Add City");
        addButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                addCity();
            }
        });

        JButton removeButton = new JButton("Remove City");
        removeButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                removeCity();
            }
        });

        JButton searchButton = new JButton("Search City");
        searchButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                searchCity();
            }
        });

        panel.add(new JLabel("City Name:"));
        panel.add(cityNameTextField);
        panel.add(new JLabel("STD Code:"));
        panel.add(stdCodeTextField);
        panel.add(addButton);
        panel.add(removeButton);
        panel.add(searchButton);

        add(panel);
    }

    private void addCity() {
        String cityName = cityNameTextField.getText().trim();
        String stdCode = stdCodeTextField.getText().trim();

        if (!cityName.isEmpty() && !stdCode.isEmpty()) {
            if (!cityCodes.containsKey(cityName)) {
                cityCodes.put(cityName, stdCode);
                JOptionPane.showMessageDialog(this, "City added
successfully.");
            } else {
                JOptionPane.showMessageDialog(this, "City already exists.
Please use a different name.");
            }
        } else {
            JOptionPane.showMessageDialog(this, "Please enter both City Name
and STD Code.");
        }

        clearFields();
    }

    private void removeCity() {
        String cityName = cityNameTextField.getText().trim();

        if (!cityName.isEmpty()) {
            if (cityCodes.containsKey(cityName)) {
                cityCodes.remove(cityName);
                JOptionPane.showMessageDialog(this, "City removed
successfully.");
            } else {
                JOptionPane.showMessageDialog(this, "City not found in the
collection.");
            }
        } else {
            JOptionPane.showMessageDialog(this, "Please enter City Name to
remove.");
        }
        clearFields();
    }

    private void searchCity() {
        String cityName = cityNameTextField.getText().trim();

        if (!cityName.isEmpty()) {
            if (cityCodes.containsKey(cityName)) {
                String stdCode = cityCodes.get(cityName);
                JOptionPane.showMessageDialog(this, "STD Code for " + cityName
+ ": " + stdCode);
            } else {
                JOptionPane.showMessageDialog(this, "City not found in the
collection.");
            }
        } else {
            JOptionPane.showMessageDialog(this, "Please enter City Name to
search.");
        }

        clearFields();
    }

    private void clearFields() {
        cityNameTextField.setText("");
        stdCodeTextField.setText("");
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                S4Q2 app = new S4Q2();
                app.setVisible(true);
            }
        });
    }
}

// Slip - 7 Q1 [JAVA]
import java.util.Random;

class NumberGenerator extends Thread {
    Random random = new Random();
    NumberProcessor processor;

    public NumberGenerator(NumberProcessor processor) {
        this.processor = processor;
    }
    public void run() {
        while(true) {
            int num = random.nextInt(100);
            processor.processNumber(num);
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                System.out.println(e);
            }
        }
    }
}

class NumberProcessor {
    public synchronized void processNumber(int num) {
        if(num % 2 == 0)
            new SquareCalculator(num).start();
        else
            new CubeCalculator(num).start();
    }
}

class SquareCalculator extends Thread {
    int num;

    public SquareCalculator(int num) {
        this.num = num;
    }

    public void run() {
        System.out.println("\nSquare of " + num + " is " + (num * num));
    }
}

class CubeCalculator extends Thread {
    int num;

    public CubeCalculator(int num) {
        this.num = num;
    }

    public void run() {
        System.out.println("\nCube of " + num + " is " + (num * num * num));
    }
}

public class S7Q1 {
    public static void main(String args[]) {
        NumberProcessor processor = new NumberProcessor();
        new NumberGenerator(processor).start();
    }
}

// Slip - 7 Q2 [JAVA]
import java.sql.*;

public class S7Q2 {
    public static void main(String args[]) {
        try {
            Class.forName("org.postgresql.Driver");
            Connection con =
DriverManager.getConnection("jdbc:postgresql://localhost:5432/ty92","ty92","ty
92");

            Statement stmt = con.createStatement();

            stmt.execute("CREATE TABLE IF NOT EXISTS Product(PID int primary
key, PNAME varchar(20), PRICE int)");
            System.out.println("Table created successfully!");

            insertRecords(stmt);
            displayRecords(stmt);

            stmt.close();

        } catch (SQLException | ClassNotFoundException e) {
            System.out.println(e);
        }
    }

    private static void insertRecords(Statement stmt) throws SQLException {
        String insertSQL = "INSERT INTO Product (PID, PNAME, PRICE) VALUES ";
        String records[] = {
            "(1, 'A', 12)",
            "(2, 'B', 32)",
            "(3, 'C', 24)",
            "(4, 'D', 14)",
            "(5, 'E', 10)"
        };

        for(String record : records)
            stmt.executeUpdate(insertSQL + record);

        System.out.println("\nRecords inserted into product table
successfully!");
```

```java
    }

    private static void displayRecords(Statement stmt) throws SQLException {
        ResultSet rs = stmt.executeQuery("SELECT * FROM Product");

        System.out.println("\n- Records from Product Table -\n");
        System.out.println("\nPID\tPNAME\tPRICE");

        while(rs.next()) {
            int pid = rs.getInt("PID");
            String pname = rs.getString("PNAME");
            int price = rs.getInt("PRICE");
            System.out.println(pid + "\t" + pname + "\t" + price);
        }

        rs.close();
    }
}

// Slip - 8 Q1 [JAVA]
import java.io.*;

class ThreadDemo extends Thread {
    String msg;
    int n;

    ThreadDemo(String msg, int n) {
        this.msg = msg;
        this.n = n;
    }

    public void run() {
        for (int i = 0; i < n; i++) {
            System.out.println(msg + ": " + (i+1));
        }
    }
}
class A1 {
    public static void main(String args[]) throws IOException {
        ThreadDemo t1 = new ThreadDemo("Thread1 -> COVID19", 10);
        ThreadDemo t2 = new ThreadDemo("Thread2 -> LOCKDOWN2020", 20);
        ThreadDemo t3 = new ThreadDemo("Thread3 -> VACCINATED21", 30);

        t1.start();
        t2.start();
        t3.start();
    }
}
```

```
<!-- Slip - 8 Q2 [JAVA] -->
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <title>Prime Number Checker</title>
</head>
<body>
    <h1>Prime Number Checker</h1>

    <!-- HTML form to enter the number -->
    <form action="" method="get">
        Enter a number: <input type="text" name="number">
        <input type="submit" value="Check">
    </form>

    <%!
        // Method to check whether a number is prime
        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;
        }
    %>

    <%-- Java code to check if the number is prime --%>
    <%
        String numberStr = request.getParameter("number");

        if (numberStr != null && !numberStr.isEmpty()) {
            int number = Integer.parseInt(numberStr);
            boolean isPrimeNumber = isPrime(number);

            if (isPrimeNumber) {
    %>
                <p style="color: red;"><%= number %> is a prime number.</p>
    <%
            } else {
    %>
                <p style="color: red;"><%= number %> is not a prime
number.</p>
```

```
    <%
            }
        } else if (numberStr != null && numberStr.isEmpty()) {
    %>
            <p style="color: red;">Please enter a number.</p>
    <%
        }
    %>
</body>
</html>

<!-- Slip - 11 Q1 [JAVA] -->
<!DOCTYPE html>
<html>
<head>
    <title>Search Customer</title>
</head>
<body>
    <h1>Search Customer</h1>
    <form action="SearchServlet" method="post">
        <label for="customerNum">Enter customer number:</label>
        <input type="number" id="customerNum" name="customerNum" required>
        <button type="submit">Search</button>
    </form>
</body>
</html>
// Slip - 11 Q2 [JAVA]
import java.sql.*;
public class S11Q2 {
    public static void main(String args[]) {
        try {
            Class.forName("org.postgresql.Driver");

            Connection con =
DriverManager.getConnection("jdbc:postgresql://localhost:5432/ty92","ty92","ty
92");
            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery("SELECT * FROM Donor");

            ResultSetMetaData rsmd = rs.getMetaData();
            int columnCount = rsmd.getColumnCount();

            System.out.println("\n- Columns in the DONOR table -\n");
            System.out.println("--------------------------------------------
");
            for(int i=1; i<=columnCount; i++) {
                System.out.println("Column Name: " + rsmd.getColumnName(i));
                System.out.println("Data Type: " + rsmd.getColumnTypeName(i));
                System.out.println("Column Size: " +
rsmd.getColumnDisplaySize(i));
                System.out.println("-------------------------------------------
--");
            }

            rs.close();
            stmt.close();
            con.close();
        } catch (SQLException | ClassNotFoundException e) {
            System.out.println(e);
        }
    }
}
<!-- Slip - 12 Q1 [JAVA] -->
<!-- Write a JSP program to check whether given number is Perfect or not. (Use
Include directive).  -->

<!-- This is one of the 3 files required for this program. [JSP] -->

<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Perfect Number Checker</title>
</head>
<body>
    <h1>Perfect Number Checker</h1>
    <form action="CheckPerfectNumber.jsp" method="post">
        Enter a number: <input type="number" name="number" required>
        <input type="submit" value="Check">
    </form>
</body>
</html>

// Slip - 12 Q2 [JAVA]
// Write a Java Program to create a PROJECT table with field’s project_id,
Project_name, Project_description, Project_Status. Insert values in the table.
Display all the details of the PROJECT table in a tabular format on the
screen.(using swing).

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

```java
class S12Q2 extends JFrame implements ActionListener {
    private JLabel pidLabel, pnameLabel, pdescLabel, pstatusLabel;
    private JTextField pidField, pnameField, pdescField, pstatusField;
    private JButton insertButton, displayButton, resetButton;
    private JPanel inputPanel, displayPanel;
    private Connection con;
    private JTable table;

    S12Q2() {
        pidLabel = new JLabel("PID:");
        pnameLabel = new JLabel("Name:");
        pdescLabel = new JLabel("Description:");
        pstatusLabel = new JLabel("Status:");

        pidField = new JTextField(20);
        pnameField = new JTextField(20);
        pdescField = new JTextField(20);
        pstatusField = new JTextField(20);

        insertButton = new JButton("INSERT");
        displayButton = new JButton("DISPLAY");
        resetButton = new JButton("RESET");

        insertButton.addActionListener(this);
        displayButton.addActionListener(this);
        resetButton.addActionListener(this);

        inputPanel = new JPanel(new FlowLayout());
        inputPanel.add(pidLabel);
        inputPanel.add(pidField);
        inputPanel.add(pnameLabel);
        inputPanel.add(pnameField);
        inputPanel.add(pdescLabel);
        inputPanel.add(pdescField);
        inputPanel.add(pstatusLabel);
        inputPanel.add(pstatusField);
        inputPanel.add(insertButton);
        inputPanel.add(displayButton);
        inputPanel.add(resetButton);

        displayPanel = new JPanel(new BorderLayout());
        displayPanel.setBorder(BorderFactory.createTitledBorder("Display"));

        setLayout(new BorderLayout());
        add(inputPanel, BorderLayout.NORTH);
        add(displayPanel, BorderLayout.CENTER);
        setSize(500, 300);
        setTitle("Project Management");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == insertButton) {
            insertRecord();
        } else if (e.getSource() == displayButton) {
            displayRecords();
        } else if (e.getSource() == resetButton) {
            resetFields();
        }
    }

    private void insertRecord() {
        int pid = Integer.parseInt(pidField.getText());
        String pname = pnameField.getText();
        String pdesc = pdescField.getText();
        String pstatus = pstatusField.getText();

        try {
            if (con == null || con.isClosed()) {
                con =
DriverManager.getConnection("jdbc:postgresql://localhost:5432/ty92", "ty92",
"ty92");
            }

            String sql = "INSERT INTO project VALUES (?, ?, ?, ?)";
            try (PreparedStatement ps = con.prepareStatement(sql)) {
                ps.setInt(1, pid);
                ps.setString(2, pname);
                ps.setString(3, pdesc);
                ps.setString(4, pstatus);
                int n = ps.executeUpdate();
                if (n != 0) {
                    JOptionPane.showMessageDialog(this, "Record inserted
successfully.");
                } else {
                    JOptionPane.showMessageDialog(this, "Failed to insert
record.");
                }
            }
        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(this, "Error: " + ex.getMessage());
        }
    }

    private void displayRecords() {
        try {
            if (con == null || con.isClosed()) {
                con =
DriverManager.getConnection("jdbc:postgresql://localhost:5432/ty92", "ty92",
"ty92");
            }

            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery("SELECT * FROM project");

            Vector<String> columnNames = new Vector<>();
            Vector<Vector<Object>> data = new Vector<>();

            ResultSetMetaData rsmd = rs.getMetaData();
            int columns = rsmd.getColumnCount();
            for (int i = 1; i <= columns; i++) {
                columnNames.addElement(rsmd.getColumnName(i));
            }

            while (rs.next()) {
                Vector<Object> row = new Vector<>();
                for (int i = 1; i <= columns; i++) {
                    row.addElement(rs.getObject(i));
                }
                data.addElement(row);
            }

            table = new JTable(data, columnNames);
            JScrollPane scrollPane = new JScrollPane(table);
            displayPanel.removeAll();
            displayPanel.add(scrollPane, BorderLayout.CENTER);
            displayPanel.revalidate();
            displayPanel.repaint();

        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(this, "Error: " + ex.getMessage());
        }
    }

    private void resetFields() {
        pidField.setText("");
        pnameField.setText("");
        pdescField.setText("");
        pstatusField.setText("");
    }
    public static void main(String args[]) {
        new S12Q2();
    }
}

// Slip - 13 Q1 [JAVA]
// Write a Java program to display information about the database and list all
the tables in the database. (Use DatabaseMetaData).

import java.sql.*;

public class S13Q1 {
    public static void main(String args[]) {
        try {
            Class.forName("org.postgresql.Driver");
            Connection con =
DriverManager.getConnection("jdbc:postgresql://localhost:5432/ty92","ty92","ty
92");
            DatabaseMetaData dbmd = con.getMetaData();

            System.out.println("\nDatabase Product Name: " +
dbmd.getDatabaseProductName());
            System.out.println("Database Product Version: " +
dbmd.getDatabaseProductVersion());
            System.out.println("Driver Name: " + dbmd.getDriverName());
            System.out.println("Driver Version: " + dbmd.getDriverVersion());

            System.out.println("\n- Tables in the Database -\n");
            ResultSet rs = dbmd.getTables(null,null,null,new
String[]{"TABLE"});
            while(rs.next())
                System.out.println(rs.getString("TABLE_NAME"));

            rs.close();
            con.close();
        } catch (SQLException | ClassNotFoundException e) {
            System.out.println(e);
        }
    }
}

// Slip - 13 Q2 [JAVA]
// Write a Java program to show lifecycle (creation, sleep, and dead) of a
thread. Program should print randomly the name of thread and value of sleep
time. The name of the thread should be hard coded through constructor. The
sleep time of a thread will be a random integer in the range 0 to 4999.
```

```java
import java.util.Random;

public class S13Q2 {
    public static void main(String args[]) {
        Thread thread = new CustomThread("CustomThread");
        thread.start();
    }

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

        public void run() {
            System.out.println(getName() + " is created.");

            Random random = new Random();
            int sleepTime = random.nextInt(5000);

            System.out.println(getName() + " will sleep for " + sleepTime + "
milliseconds.");

            try {
                Thread.sleep(sleepTime);
            } catch (InterruptedException e) {
                System.out.println(e);
            }

            System.out.println(getName() + " is dead.");
        }
    }
}

// Slip - 14 Q1 [JAVA]
//  Write a Java program for a simple search engine. Accept a string to be
searched. Search  the string in all text files in the current folder. Use a
separate thread for each file. The result should display the filename and line
number where the string is found.

import java.io.*;
import java.util.*;
import java.util.concurrent.*;

public class S14Q1 {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);

        System.out.print("\nEnter the string to search: ");
        String searchString = sc.nextLine();
        sc.close();

        File folder = new File(System.getProperty("user.dir"));
        File files[] = folder.listFiles();

        if(files != null) {
            ExecutorService executor = 
Executors.newFixedThreadPool(files.length);
            for(File file : files) {
                if(file.isFile() && file.getName().endsWith(".txt")) {
                    executor.execute(new SearchTask(file, searchString));
                }
            }
            executor.shutdown();
        }
    }

    private static class SearchTask implements Runnable {
        private File file;
        private String searchString;

        public SearchTask(File file, String searchString) {
            this.file = file;
            this.searchString = searchString;
        }

        public void run() {
            searchInFile(file);
        }

        private void searchInFile(File file) {
            try {
                BufferedReader reader = new BufferedReader(new 
FileReader(file));
                String line;
                int lineNumber = 0;
                while((line = reader.readLine()) != null) {
                    lineNumber++;
                    if(line.contains(searchString)) {
                        System.out.println("\nFound in file: " + 
file.getName() + ", Line: " + lineNumber);
                    }
                }
                reader.close();
            } catch (Exception e) {
                System.out.println(e);
            }
        }
    }
}

<!-- Slip - 14 Q1 [JAVA] -->
<!-- Write a JSP program to calculate sum of first and last digit of a given
number. Display sum in Red Color with font size 18.  -->

<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8" %>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Sum of first and last digit</title>
</head>
<body>
    <h2>Calculate Sum of First and Last Digits</h2>
    <form action="" method="post">
        Enter a number: <input type="text" name="number"><br>
        <input type="submit" value="Calculate">
    </form>

    <%
        String numberStr = request.getParameter("number");
        if(numberStr != null && !numberStr.isEmpty()) {
            int number = Integer.parseInt(numberStr);
            int firstDigit = Character.getNumericValue(numberStr.charAt(0));
            int lastDigit = number % 10;
            int sum = firstDigit + lastDigit;
    %>
    <p style="color: red; font-size: 18px;">Sum of first and last digit of <%=
number %>: <%= sum %></p>
    <% } %>
</body>
</html>

// Slip - 15 Q1 [JAVA]
// Write a java program to display name and priority of a Thread.

public class S15Q1 {
    public static void main(String args[]) {
        Thread thread = Thread.currentThread();
        System.out.println("\nThread Name: " + thread.getName());
        System.out.println("Thread Priority: " + thread.getPriority());
    }
}
```

```java
// Slip - 15 Q2 [JAVA]
// Write a SERVLET program which counts how many times a user has visited a
web page. If user is visiting the page for the first time, display a welcome
message. If the user is revisiting the page, display the number of times
visited. (Use Cookie)

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class S15Q2 extends HttpServlet {
    public void doGet(HttpServletRequest req, HttpServletResponse res) throws
IOException, ServletException {
        res.setContentType("text/html");

        int visitCount = 0;
        Cookie[] cookies = req.getCookies();
        if (cookies != null) {
            for (Cookie cookie : cookies) {
                if (cookie.getName().equals("visitCount")) {
                    visitCount = Integer.parseInt(cookie.getValue());
                    break;
                }
            }
        }

        PrintWriter out = res.getWriter();
        out.println("<html>");
        out.println("<body>");

        if (visitCount == 0)
            out.println("<h1>Welcome! This is your first visit to the
page.</h1>");
        else
            out.println("<h1>Welcome back! You've visited this page " +
visitCount + " times.</h1>");

        visitCount++;
        Cookie visitCookie = new Cookie("visitCount",
String.valueOf(visitCount));
        res.addCookie(visitCookie);

        out.println("<form method=\"post\">");
        out.println("<input type=\"submit\" value=\"Refresh\">");
        out.println("</form>");
        out.println("</body>");
        out.println("</html>");
    }
    public void doPost(HttpServletRequest req, HttpServletResponse res) throws
IOException, ServletException {
        doGet(req, res);
    }
}

// Slip - 16 Q1 [JAVA]
// Write a java program to create a TreeSet, add some colors (String) and
print out the content of TreeSet in ascending order.

import java.util.*;

public class S16Q1 {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);

        TreeSet<String> colors = new TreeSet<>();

        System.out.print("\nEnter the number of colors: ");
        int n = sc.nextInt();
        sc.nextLine();

        for(int i=0; i<n; i++) {
            System.out.print("Enter color " + (i+1) + ": ");
            colors.add(sc.nextLine());
        }

        System.out.println("\nTreeSet in ascending order is: " + colors);
        sc.close();
    }
}

// Slip - 16 Q2 [JAVA]
// Write a Java program to accept the details of Teacher (TNo,  TName,
Subject). Insert at least 5 Records into Teacher Table and display the details
of Teacher who is teaching “JAVA” Subject.  (Use PreparedStatement Interface)

import java.sql.*;

public class S16Q2 {
    public static void main(String args[]) {
        try {
            Class.forName("org.postgresql.Driver");
            Connection con =
DriverManager.getConnection("jdbc:postgresql://localhost:5432/ty92","ty92","ty
92");
```

```
            Statement stmt = con.createStatement();
            PreparedStatement pstmt = null;

            stmt.executeUpdate("CREATE TABLE IF NOT EXISTS Teacher (TNO int 
primary key, TNAME varchar(20), SUBJECT varchar(20))");

            System.out.println("Table exists / created!");
            System.out.println("Inserting values into the table...");

            pstmt = con.prepareStatement("INSERT INTO 
Teacher(TNO,TNAME,SUBJECT) VALUES (?,?,?)");

            insertTeacher(pstmt, 101, "Taskar", "JAVA");
            insertTeacher(pstmt, 102, "Mahale", "DSA");
            insertTeacher(pstmt, 103, "Deore", "C");
            insertTeacher(pstmt, 104, "Patil", "CN");
            insertTeacher(pstmt, 105, "Kapse", "OS");

            System.out.println("\nRecords inserted successfully!");

            System.out.println("\nDisplaying details of teachers teaching 
'JAVA' subject...");

            pstmt = con.prepareStatement("SELECT * FROM Teacher WHERE SUBJECT 
= ?");
            pstmt.setString(1,"JAVA");

            ResultSet rs = pstmt.executeQuery();

            while(rs.next()) {
                int tno = rs.getInt("TNO");
                String tname = rs.getString("TNAME");
                String sub = rs.getString("SUBJECT");

                System.out.println("\nTeacher No.: " + tno);
                System.out.println("Teacher Name: " + tname);
                System.out.println("Subject: " + sub);
            }

            rs.close();
            pstmt.close();
            con.close();
        } catch (SQLException | ClassNotFoundException e) {
            System.out.println(e);
        }
    }
    private static void insertTeacher(PreparedStatement pstmt, int tno, String
tname, String sub) throws SQLException {
        pstmt.setInt(1, tno);
        pstmt.setString(2, tname);
        pstmt.setString(3, sub);
        pstmt.executeUpdate();
    }
}

// Slip - 17 Q1 [JAVA]
// Write a java program to accept ‘N’ integers from a user. Store and display
integers in sorted order having proper collection class. The collection should
not accept duplicate elements.

import java.util.*;

public class S17Q1 {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);

        TreeSet<Integer> set = new TreeSet<>();

        System.out.print("\nEnter the number of integers: ");
        int n = sc.nextInt();

        for(int i=0; i<n; i++) {
            System.out.print("Enter integer " + (i+1) + ": ");
            set.add(sc.nextInt());
        }

        System.out.println("\nSorted set without duplicates:-");
        for(int num : set)
            System.out.println(num);

        sc.close();
    }
}

// Slip - 17 Q2 [JAVA]
// Write a Multithreading program in java to display the number’s between 1 to
100 continuously in a TextField by clicking on button. (Use Runnable
Interface).

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

public class S17Q2 extends JFrame {
```

```java
    JTextField textField;
    JButton startBtn;

    public S17Q2() {
        setTitle("Number Display");
        setSize(450,300);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLayout(new FlowLayout());

        textField = new JTextField(10);
        add(textField);

        startBtn = new JButton("Start");
        startBtn.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                startDisplay();
            }
        });
        add(startBtn);

        setVisible(true);
    }

    private void startDisplay() {
        Thread displayThread = new Thread(new DisplayRunnable());
        displayThread.start();
    }
    private class DisplayRunnable implements Runnable {
        public void run() {
            for(int i=1; i<=100; i++) {
                textField.setText(Integer.toString(i));
                try {
                    Thread.sleep(100);
                } catch (InterruptedException e) {
                    System.out.println(e);
                }
            }
            textField.setText("");
        }
    }

    public static void main(String args[]) {
        new S17Q2();
    }
}

// Slip - 18 Q1 [JAVA]
// Repeated from Slip - 15 Q1
```

```
<!-- Slip - 18 Q2 [JAVA] -->
<!-- Write a SERVLET program in java to accept details of student (SeatNo,
Stud_Name, Class, Total_Marks). Calculate percentage and grade obtained and
display details on page. -->

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Details Form</title>
</head>
<body>
    <h2>Enter Student Details</h2>
    <form action="S18Q2" method="post">
        <label for="seat_no">Seat No:</label>
        <input type="text" id="seat_no" name="seat_no" required><br><br>

        <label for="name">Name:</label>
        <input type="text" id="name" name="name" required><br><br>

        <label for="class">Class:</label>
        <input type="text" id="class" name="class" required><br><br>

        <label for="total_marks">Total Marks:</label>
        <input type="text" id="total_marks" name="total_marks"
required><br><br>

        <input type="submit" value="Submit">
    </form>
</body>
</html>
// Slip - 18 Q2 [JAVA]
// Write a SERVLET program in java to accept details of student (SeatNo,
Stud_Name, Class, Total_Marks). Calculate percentage and grade obtained and
display details on page.

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class S18Q2 extends HttpServlet {
    public void doPost(HttpServletRequest req, HttpServletResponse res) throws
IOException, ServletException {
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();

        int seatNo = Integer.parseInt(req.getParameter("seat_no"));
        String name = req.getParameter("name");
        String stu_class= req.getParameter("class");
        int totalMarks = Integer.parseInt(req.getParameter("total_marks"));

        double perc = (totalMarks / 500.0) * 100;

        String grade;
        if (perc >= 90) {
            grade = "A+";
        } else if (perc >= 80) {
            grade = "A";
        } else if (perc >= 70) {
            grade = "B";
        } else if (perc >= 60) {
            grade = "C";
        } else if (perc >= 50) {
            grade = "D";
        } else {
            grade = "Fail!";
        }

        out.println("<html><head><title>Student
Details</title></head><body>");
        out.println("<h2>Student Details</h2>");
        out.println("<p>Seat No.: " + seatNo + "</p>");
        out.println("<p>Name: " + name + "</p>");
        out.println("<p>Class: " + stu_class + "</p>");
        out.println("<p>Total Marks: " + totalMarks + "</p>");
        out.println("<p>Percentage: " + perc + "</p>");
        out.println("<p>Grade: " + grade + "</p>");
        out.println("</body></html>");

        out.close();
    }
}

// Slip - 19 Q1 [JAVA]
// Write a java program to accept ‘N’ Integers from a user store them into
LinkedList Collection and display only negative integers.

import java.util.*;

public class S19Q1 {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);

        System.out.print("\nEnter the number of integers: ");
        int n = sc.nextInt();
        LinkedList<Integer> list = new LinkedList<>();

        for(int i=0; i<n; i++) {
            System.out.print("Enter integer " + (i+1) + ": ");
            list.add(sc.nextInt());
        }

        System.out.println("\nNegative integers from list:-");
        for(int num : list) {
            if(num < 0) {
                System.out.println(num);
            }
        }

        sc.close();
    }
}

<!-- Slip - 19 Q2 [JAVA] -->
<!-- Write a SERVLET application to accept username and password, search them
into database, if found then display appropriate message on the browser
otherwise display error message. -->

<!DOCTYPE html>
<html>
    <head>
        <title>Validate User</title>
    </head>
    <body>
        <form action="UserValidation" method="post">
            <label for="username">Enter username:</label>
            <input type="text" id="username" name="username" required>

            <label for="password">Enter password:</label>
            <input type="password" id="password" name="password" required>

            <input type="submit" value="Submit">
        </form>
    </body>
</html>
// Slip - 21 Q1 [JAVA]
// Write a java program to accept 'N' Subject Names from a user store them
into LinkedList Collection and Display them by using Iterator interface.

import java.util.*;

public class S21Q1 {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        LinkedList<String> list = new LinkedList<>();

        System.out.print("\nEnter the number of subjects: ");
        int n = sc.nextInt();
        sc.nextLine();

        for(int i=0; i<n; i++) {
            System.out.print("Enter subject " + (i+1) + ": ");
            list.add(sc.nextLine());
        }

        System.out.println("\nList items:-");

        Iterator<String> itr = list.iterator();
        while(itr.hasNext()) {
            System.out.println(itr.next());
        }
        sc.close();
    }
}

// Slip - 21 Q2 [JAVA]
// Write a java program to solve producer consumer problem in which a producer
produces a value and consumer consume the value before producer generate the
next value. (Hint: use thread synchronization)

class Buffer {
    private String data;
    private boolean produced;

    Buffer() {
        this.data = null;
        this.produced = false;
    }

    public synchronized void produce(String item) throws InterruptedException
{
        // Wait until the previous value is consumed
        while (produced)
            wait();

        // Produce the new value
        data = item;
        produced = true;
        System.out.println("Produced: " + item);
        // Notify the consumer
        notify();
    }

    public synchronized String consume() throws InterruptedException {
        // Wait until a new value is produced
        while (!produced)
            wait();

        // Consume the produced value
        String consumedItem = data;
        produced = false;
        System.out.println("Consumed: " + consumedItem);

        // Notify the producer
        notify();

        return consumedItem;
    }
}

class Producer extends Thread {
    private String msg;
    private Buffer buffer;
    private int count;

    Producer(String msg, Buffer buffer, int count) {
        this.msg = msg;
        this.buffer = buffer;
        this.count = count;
    }

    public void run() {
        try {
            for (int i = 0; i < count; i++) {
                buffer.produce(msg);
                Thread.sleep(1000);
            }
        } catch (InterruptedException e) {
            System.out.println(e);
        }
    }
}

class Consumer extends Thread {
    private Buffer buffer;
    private int count;
    Consumer(Buffer buffer, int count) {
        this.buffer = buffer;
        this.count = count;
    }

    public void run() {
        try {
            for (int i = 0; i < count; i++) {
                buffer.consume();
                Thread.sleep(1000);
            }
        } catch (InterruptedException e) {
            System.out.println(e);
        }
    }
}

public class S21Q2 {
    public static void main(String args[]) {
        Buffer buffer = new Buffer();
        Producer producer = new Producer("Hello!", buffer, 6);
        Consumer consumer = new Consumer(buffer, 6);

        producer.start();
        consumer.start();
    }
}

// Slip - 22 Q1 [JAVA]
// Write a Menu Driven program in Java for the following: Assume Employee
table with attributes (ENo, EName, Salary) is already created. 1. Insert 2.
Update 3. Display 4. Exit.

import java.sql.*;

public class S22Q1 {
    static final String JDBC_DRIVER = "org.postgresql.Driver";
    static final String DB_URL = "jdbc:postgresql://localhost:5432/ty92";
    static final String USER = "ty92";
    static final String PASS = "ty92";

    public static void main(String[] args) {
        Connection conn = null;
        Statement stmt = null;
        try {
            Class.forName(JDBC_DRIVER);
            conn = DriverManager.getConnection(DB_URL, USER, PASS);
            stmt = conn.createStatement();
            boolean exit = false;
            while (!exit) {
                System.out.println("\nEmployee Management System");
                System.out.println("1. Insert");
                System.out.println("2. Update");
                System.out.println("3. Display");
                System.out.println("4. Exit");
                System.out.print("Enter your choice: ");

                int choice = Integer.parseInt(System.console().readLine());

                switch (choice) {
                    case 1:
                        insertEmployee(stmt);
                        break;
                    case 2:
                        updateEmployee(stmt);
                        break;
                    case 3:
                        displayEmployees(stmt);
                        break;
                    case 4:
                        exit = true;
                        break;
                    default:
                        System.out.println("Invalid choice! Please enter a
number between 1 and 4.");
                }
            }

            stmt.close();
            conn.close();
        } catch (SQLException se) {
            se.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (stmt != null) stmt.close();
            } catch (SQLException se2) {
            }
            try {
                if (conn != null) conn.close();
            } catch (SQLException se) {
                se.printStackTrace();
            }
        }
```

```java
    }

    static void insertEmployee(Statement stmt) throws SQLException {
        System.out.println("\nInsert Employee");
        System.out.print("Enter Employee Number: ");
        int eno = Integer.parseInt(System.console().readLine());
        System.out.print("Enter Employee Name: ");
        String ename = System.console().readLine();
        System.out.print("Enter Designation: ");
        String designation = System.console().readLine();
        System.out.print("Enter Salary: ");
        double salary = Double.parseDouble(System.console().readLine());

        String sql = "INSERT INTO Employee (eno, ename, designation, salary)
VALUES (" + eno + ", '" + ename + "', '" + designation + "', " + salary + ")";
        stmt.executeUpdate(sql);
        System.out.println("Employee inserted successfully.");
    }

    static void updateEmployee(Statement stmt) throws SQLException {
        System.out.println("\nUpdate Employee");
        System.out.print("Enter Employee Number: ");
        int eno = Integer.parseInt(System.console().readLine());
        System.out.print("Enter new Salary: ");
        double salary = Double.parseDouble(System.console().readLine());

        String sql = "UPDATE Employee SET Salary=" + salary + " WHERE eno=" +
eno;
        int rowsAffected = stmt.executeUpdate(sql);
        if (rowsAffected > 0)
            System.out.println("Employee updated successfully.");
        else
            System.out.println("Employee not found.");
    }

    static void displayEmployees(Statement stmt) throws SQLException {
        System.out.println("\nEmployee List");
        String sql = "SELECT * FROM Employee";
        ResultSet rs = stmt.executeQuery(sql);

        while (rs.next()) {
            int eno = rs.getInt("eno");
            String ename = rs.getString("ename");
            String designation = rs.getString("designation");
            double salary = rs.getDouble("salary");
            System.out.println("Employee Number: " + eno + ", Employee Name: "
+ ename + ", Designation: " + designation + ", Salary: " + salary);
        }
        rs.close();
    }
}

<!-- Slip - 22 Q2 [JAVA] -->
<!-- Write a JSP program which accepts UserName in a TextBox and greets the
user according to the time on server machine. -->

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ page import="pkg.GreetingService" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Greeting</title>
</head>
<body>
    <h2>Greeting</h2>

    <form action="" method="post">
        Enter your name: <input type="text" name="username"><br>
        Enter your password: <input type="password" name="password"><br>
        <input type="submit" value="Submit">
    </form>

    <%
        request.setCharacterEncoding("UTF-8");
        String username = request.getParameter("username");
        String password = request.getParameter("password");

        // Check if username and password are not null or empty
        if(username != null && !username.isEmpty() && password != null &&
!password.isEmpty()) {
            String greeting = GreetingService.getGreeting(username);
            out.println("<p>" + greeting + "</p>");
        }
    %>
</body>
</html>

// Slip - 23 Q1 [JAVA]
// Write a java program to accept a String from a user and display each vowel
from a String after every 3 seconds.

import java.util.*;
```

```java
class VowelThread extends Thread {
    private String inputString;

    public VowelThread(String inputString) {
        this.inputString = inputString;
    }

    public void run() {
        try {
            for(int i=0; i<inputString.length(); i++) {
                char ch = inputString.charAt(i);
                if(isVowel(ch)) {
                    System.out.println(ch);
                    Thread.sleep(3000);
                }
            }
        } catch (InterruptedException e) {
            System.out.println(e);
        }
    }

    private boolean isVowel(char ch) {
        ch = Character.toLowerCase(ch);
        return ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u';
    }
}

public class S23Q1 {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.print("\nEnter a string: ");
        String inputString = sc.nextLine();

        VowelThread thread = new VowelThread(inputString);
        thread.start();
        sc.close();
    }
}

// Slip - 23 Q2 [JAVA]
// Write a java program to accept ‘N’ student names through command line,
store them into the appropriate Collection and display them by using Iterator
and ListIterator interface.

import java.util.*;

public class S23Q2 {
    public static void main(String args[]) {
        List<String> studentNames = new ArrayList<>();

        for(String arg : args)
            studentNames.add(arg);

        System.out.println("\nStudent names using Iterator:-");
        Iterator<String> itr = studentNames.iterator();
        while(itr.hasNext())
            System.out.println(itr.next());

        System.out.println("\nStudent names using ListIterator:-");
        ListIterator<String> listItr = studentNames.listIterator();
        while(listItr.hasNext())
            System.out.println(listItr.next());
    }
}

// Slip - 24 Q1 [JAVA]
// Write a java program to scroll the text from left to right continuously.

import javax.swing.*;

public class S24Q1 extends JFrame implements Runnable {
    private JLabel label;

    public S24Q1() {
        setTitle("Text Scrolling");
        setSize(400,100);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        label = new JLabel("This is the scrolling text!");
        add(label);

        Thread thread = new Thread(this);
        thread.start();

        setVisible(true);
    }

    public void run() {
        try {
            while(true) {
                Thread.sleep(100);
                String text = label.getText();
                text = text.substring(1) + text.charAt(0);
                label.setText(text);
            }
        } catch (InterruptedException e) {
            System.out.println(e);
        }
    }

    public static void main(String args[]) {
        new S24Q1();
    }
}

<!-- Slip - 24 Q2 [JAVA] -->
<!-- Write a JSP script to accept username and password from user, if they are
same then display “Login Successfully” message in Login.html file, otherwise
display “Login Failed” Message in Error.html file.  -->

<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8" %>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>Login</title>
    </head>
    <body>
        <%
            String username = request.getParameter("username");
            String password = request.getParameter("password");

            if(username != null && password != null &&
username.equals(password)) {
        %>
                <h1>Login Successfully</h1>
        <%
            } else {
                response.sendRedirect("Error.html");
            }
        %>
    </body>
</html>

// Slip - 26 Q1 [JAVA]
// Write a Java program to delete the details of given employee (ENo EName
Salary). Accept employee ID through command line. (Use PreparedStatement
Interface)

import java.sql.*;

public class S26Q1 {
    public static void main(String args[]) {
        int empId = Integer.parseInt(args[0]);

        try {
            Class.forName("org.postgresql.Driver");
            Connection con =
DriverManager.getConnection("jdbc:postgresql://localhost:5432/ty92","ty92","ty
92");

            PreparedStatement pstmt = con.prepareStatement("DELETE FROM
Employee WHERE eno = ?");
            pstmt.setInt(1, empId);

            int rowAffected = pstmt.executeUpdate();

            if(rowAffected > 0)
                System.out.println("\nDetails of employee with ID " + empId +
" deleted successfully!");
            else
                System.out.println("Employee with ID " + empId + " not
found.");
        } catch(SQLException | ClassNotFoundException e) {
            System.out.println(e);
        }
    }
}

<!-- Slip - 26 Q2 [JAVA] -->
<!-- Repeated from Slip - 14 Q2 -->

// Slip - 27 Q1 [JAVA]
// Write a Java Program to display the details of College (CID, CName,
address, Year) on JTable.

import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.util.ArrayList;

public class S27Q1 extends JFrame {

    public S27Q1() {
        setTitle("College Details");
        setSize(600, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        initComponents();
    }
```

```java
    private void initComponents() {
        // Sample data for colleges
        ArrayList<College> colleges = new ArrayList<>();
        colleges.add(new College(1, "ABC College", "123 Main St", 2000));
        colleges.add(new College(2, "XYZ College", "456 Elm St", 1995));
        colleges.add(new College(3, "PQR College", "789 Oak St", 2010));

        // Create JTable with DefaultTableModel
        String[] columnNames = {"CID", "CName", "Address", "Year"};
        DefaultTableModel model = new DefaultTableModel(columnNames, 0);
        for (College college : colleges) {
            Object[] rowData = {college.getCID(), college.getCName(),
college.getAddress(), college.getYear()};
            model.addRow(rowData);
        }

        JTable table = new JTable(model);

        // Add JTable to JScrollPane
        JScrollPane scrollPane = new JScrollPane(table);

        getContentPane().add(scrollPane, BorderLayout.CENTER);
    }

    public static void main(String[] args) {
        new S27Q1().setVisible(true);
    }
}

class College {
    private int CID;
    private String CName;
    private String address;
    private int year;

    public College(int CID, String CName, String address, int year) {
        this.CID = CID;
        this.CName = CName;
        this.address = address;
        this.year = year;
    }

    public int getCID() {
        return CID;
    }

    public String getCName() {
        return CName;
    }

    public String getAddress() {
        return address;
    }

    public int getYear() {
        return year;
    }
}

// Slip - 27 Q2 [JAVA]
// Write a SERVLET program to change inactive time interval of session.

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class S27Q2 extends HttpServlet {

    protected void doGet(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {

        // Get the current session or create a new one if it doesn't exist
        HttpSession session = request.getSession();

        // Set the inactive time interval of the session to 5 minutes (300
seconds)
        session.setMaxInactiveInterval(300);

        // Set response content type
        response.setContentType("text/html");

        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Session Timeout Interval Changed</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Session Timeout Interval Changed</h1>");
        out.println("<p>The inactive time interval of the session has been
changed to 5 minutes.</p>");
        out.println("</body>");
        out.println("</html>");
    }
}
```

```
<!-- Slip - 28 Q1 [JAVA] -->
<!-- Write a JSP script to accept a String from a user and display it in
reverse order. -->

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Reverse String</title>
</head>
<body>
    <h2>Enter a String:</h2>
    <form action="" method="post">
        <input type="text" name="inputString">
        <input type="submit" value="Reverse">
    </form>
    <%
        // Get the input string from the request parameter
        String inputString = request.getParameter("inputString");

        // Check if the input string is not null or empty
        if (inputString != null && !inputString.isEmpty()) {
            // Reverse the input string
            StringBuilder reversedString = new
StringBuilder(inputString).reverse();

            // Display the reversed string
    %>
            <h2>Reversed String:</h2>
            <p><%= reversedString.toString() %></p>
    <%
        }
    %>
</body>
</html>

// Slip - 28 Q2 [JAVA]
// Write a java program to display name of currently executing Thread in
multithreading.

public class S28Q2 {
    public static void main(String[] args) {
        // Create and start a new thread
        Thread thread = new Thread(new MyRunnable());
        thread.start();
        // Display the name of the main thread
        System.out.println("Main thread name: " +
Thread.currentThread().getName());
    }
}

class MyRunnable implements Runnable {
    public void run() {
        System.out.println("Currently executing thread name: " +
Thread.currentThread().getName());
    }
}

// Slip - 29 Q1 [JAVA]
// Write a Java program to display information about all columns in the DONAR
table using ResultSetMetaData.

import java.sql.*;

public class S29Q1 {
    public static void main(String args[]) {
        try {
            Class.forName("org.postgresql.Driver");
            Connection con =
DriverManager.getConnection("jdbc:postgresql://localhost:5432/ty92","ty92","ty
92");
            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery("SELECT * FROM Donor");
            ResultSetMetaData rsmd = rs.getMetaData();

            for(int i = 1; i <= rsmd.getColumnCount(); i++) {
                System.out.println("Column Name: " + rsmd.getColumnName(i));
                System.out.println("Data Type: " + rsmd.getColumnTypeName(i));
                System.out.println("Column Type: " + rsmd.getColumnType(i));
                System.out.println("------------------------------------------
--------");
            }
        } catch (SQLException | ClassNotFoundException e) {
            System.out.println(e);
        }
    }
}

// Slip - 29 Q2 [JAVA]
// Write a Java program to create LinkedList of integer objects and perform
the following:
// i. Add element at first position
```

```java
// ii. Delete last element
// iii. Display the size of link list

import java.util.*;

public class S29Q2 {
    public static void main(String args[]) {
        LinkedList<Integer> list = new LinkedList<>();

        list.add(20);
        list.add(30);
        System.out.println("\nOriginal Liked List: " + list);

        list.addFirst(10);
        System.out.println("\nLiked List after adding element at first
position: " + list);

        list.removeLast();
        System.out.println("\nLiked List after deleting the last element: " +
list);
        System.out.println("\nSize of the Linked List: " + list.size());
    }
}

// Slip - 30 Q1 [JAVA]
// Write a java program for the implementation of synchronization.

class SharedResource {
    private int value = 0;

    public synchronized void increment() {
        value++;
        System.out.println("Value incremented to: " + value);
    }

    public synchronized void decrement() {
        value--;
        System.out.println("Value decremented to: " + value);
    }
}

class IncrementThread extends Thread {
    private SharedResource res;

    public IncrementThread(SharedResource res) {
        this.res = res;
    }
    public void run() {
        for(int i=0; i<5; i++) {
            res.increment();
        }
    }
}

class DecrementThread extends Thread {
    private SharedResource res;

    public DecrementThread(SharedResource res) {
        this.res = res;
    }

    public void run() {
        for(int i=0; i<5; i++) {
            res.decrement();
        }
    }
}

public class S30Q1 {
    public static void main(String args[]) {
        SharedResource res = new SharedResource();

        IncrementThread Thread1 = new IncrementThread(res);
        DecrementThread Thread2 = new DecrementThread(res);

        Thread1.start();
        Thread2.start();
    }
}

// Slip - 30 Q2 [JAVA]
import java.sql.*;
import java.awt.*;
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
public class S30Q2 extends JFrame {
    private JTable table;
    public S30Q2() {
        super("Teacher Information");
        // Create a panel for holding the table
        JPanel panel = new JPanel(new BorderLayout());
        getContentPane().add(panel);

        // Create a scroll pane for the table
        JScrollPane scrollPane = new JScrollPane();
        panel.add(scrollPane, BorderLayout.CENTER);

        // Create a table model
        DefaultTableModel model = new DefaultTableModel();

        // Set the column names
        model.setColumnIdentifiers(new String[]{"Teacher ID", "Name",
"Subject"});

        // Create the table with the model
        table = new JTable(model);
        // Set table properties
        table.setAutoResizeMode(JTable.AUTO_RESIZE_ALL_COLUMNS);
        table.setFillsViewportHeight(true);
        // Add the table to the scroll pane
        scrollPane.setViewportView(table);
        // Populate the table
        try {
            Class.forName("org.postgresql.Driver");
            Connection con =
DriverManager.getConnection("jdbc:postgresql://localhost:5432/ty92","ty92","ty
92");
            Statement stmt =
con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY);
            ResultSet rs = stmt.executeQuery("SELECT * FROM Teacher");
            // Populate the table model with data from the ResultSet
            while (rs.next()) {
                int no = rs.getInt("tno");
                String name = rs.getString("tname");
                String subject = rs.getString("subject");
                model.addRow(new Object[]{no, name, subject});
            }
        } catch (SQLException | ClassNotFoundException e) {
            System.out.println(e);
        }
        // Set frame properties
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(600, 400);
        setLocationRelativeTo(null);
        setVisible(true);
    }

    public static void main(String[] args) {
        new S30Q2();
    }
}
```