

PROGRAM – 1

Objective:-

Write HTML/Java scripts to display your CV in the navigator, your institute website, your Department Website and the Tutorial website for specific subjects.

Program:-

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CV – JAIMANI CHOUDHARY</title>
  <!-- CSS Styles -->
  <style>
body {
  font-family: Arial, sans-serif;
margin: 0;          padding: 0;
                    background-color: #f4f4f4;
                    }
  .container {      width: 80%;          margin:
20px auto;          background-color: #fff;
padding: 20px;       border-radius: 5px;      box-
shadow: 0 0 10px rgba(0, 0, 0, 0.1);      text-align: left;
                    }    h1 {
                        color: #333;          font-size: 36px;
margin-bottom: 10px;
                    }
                    h2 {
                        color: #333;
                    }
                    .section {
                        margin-bottom: 20px;
                    }
                    .section h2 {
margin-bottom: 10px;          border-bottom:
1px solid #ccc;              padding-bottom: 5px;
                    }
                    .section p {
margin: 0;
                    }
```

```
</style>
</head>
<body>
  <div class="container">
    <!-- Header Section -->
    <header>
      <h1>JAIMANI CHOUDHARY</h1>
      <p>Email: xyz@example.com | Phone: +1234567890 |
        LinkedIn: linkedin.com/in/username</p>
    </header>

    <hr>

    <!-- Education Section -->
    <div class="section">
      <h2>Education</h2>
      <p>Bachelor of Technology in Computer Science And Engineering (CSE)</p>
      <ul>
        <li>GL Bajaj Institute of Technology and Management,
          2021-2025</li>
        <li>CGPA:</li>
        <ul>
          <li>Year 1: 7.3</li>
          <li>Year 2: 7.1</li>
        </ul>
        </ul>
        <p>Senior Secondary (12th Grade), XYZ School, 2020</p>
        <p>High School (10th Grade), XYZ School, 2018</p>
      </div>

      <!-- Work Experience Section -->
      <div class="section">
        <h2>Work Experience</h2>
        <p>Intern, Cyber Security, Palo Alto Networks, 2023</p>
        <ul>
          <li>Gained hands-on experience in network security and
            threat detection.</li>
          <li>Assisted in analyzing and mitigating security threats.</li>
        </ul>
        <p>Intern, Machine Learning, Bharat Intern, 2023</p>
      </div>
    </div>
  </body>
</html>
```

```
        <ul>
            <li>Worked on various machine learning projects and
            algorithms.</li>
            <li>Developed models for predictive analysis and pattern
            recognition.</li>
        </ul>
    </div>

    <!-- Skills Section -->
    <div class="section">
        <h2>Skills</h2>
        <ul>
            <li>Programming Languages: JavaScript, Python, Java</li>
            <li>Web Development: HTML, CSS, React.js</li>
            <li>Database Management: SQL, MongoDB</li>
        </ul>
    </div>

    <!-- Projects Section -->
    <div class="section">
        <h2>Projects</h2>
        <p>Project 1: IPL Score Prediction</p>
        <p>Project 2: House Prediction</p>
    </div>
</div>
</body>
</html>
```

Output:-

JAIMANI CHOUDHARY

Email: xyz@example.com | Phone: +1234567890 | LinkedIn: linkedin.com/in/username

Education

Bachelor of Technology in Computer Science And Engineering (CSE)

- GL Bajaj Institute of Technology and Management, 2021-2025
- CGPA:
 - Year 1: 7.3
 - Year 2: 7.1

Senior Secondary (12th Grade), XYZ School, 2020
High School (10th Grade), XYZ School, 2018

Work Experience

Intern, Cyber Security, Palo Alto Networks, 2023

- Gained hands-on experience in network security and threat detection.
- Assisted in analyzing and mitigating security threats.

Intern, Machine Learning, Bharat Intern, 2023

- Worked on various machine learning projects and algorithms.
- Developed models for predictive analysis and pattern recognition.

Skills

- Programming Languages: JavaScript, Python, Java
- Web Development: HTML, CSS, React.js
- Database Management: SQL, MongoDB

Projects

Project 1: IPL Score Prediction
Project 2: House Prediction

PROGRAM – 2

Objective:-

Design an HTML form for keeping student records and validate it using Javascript. And send it to store on a database server like SQL, Oracle or MS Access.

Program:-

File Name: 'main.html'

```
<html>
<head>
  <title>Student Registration Form</title>
  <style> form {
width: 50%;    margin:0 auto;
  }
  </style>
</head>
<body>
  <form action="f1.html" method="post" onsubmit="return validateForm()">
    <fieldset>
      <legend>STUDENT REGISTRATION FORM</legend>
      <label for="firstName">First Name:</label>
      <input type="text" name="firstName" id="firstName" /><br /><br />
      <label for="lastName">Last Name:</label>
      <input type="text" name="lastName" id="lastName" /><br /><br />
      <label for="dob">DATE OF BIRTH:</label>
      <input type="date" name="dob" id="dob" /><br /><br />
      <label>GENDER:</label>
      <input type="radio" value="male" name="gender" id="male" />
      <label for="male">Male</label>
      <input type="radio" value="female" name="gender" id="female" />
      <label for="female">Female</label> <br /><br />
      <label for="course">COURSE:</label>
      <select name="course" id="course">
        <option value="B.Tech">B.Tech</option>
        <option value="MBA">MBA</option>
      </select>
    </fieldset>
  </form>
</body>
</html>
```

```

        <option value="Other">Other</option>
    </select><br /><br />
    <label for="branch">BRANCH:</label>
    <input type="text" name="branch" id="branch" /><br /><br />
    <label for="phone">PHONE:</label>
    <input type="tel" name="phone" id="phone" maxlength="10"
        placeholder="phone" /><br /><br />
    <label for="email">EMAIL:</label>
    <input type="email" name="email" id="email" placeholder="email" /><br /><br />
    <label for="address">ADDRESS:</label>
    <textarea name="address" id="address" rows="4" cols="50"
        placeholder="address"></textarea><br /><br />
    <input type="submit" value="Submit" />
    <input type="reset" value="Reset" />
</fieldset>
</form>
<script src="index.js"></script>
</body>
</html>

```

```

File Name: 'index.js' function validateForm() { var
firstName = document.getElementById("firstName");
    var lastName = document.getElementById("lastName");        var
dob = document.getElementById("dob");        var gender =
document.querySelector('input[name="gender"]:checked');        var course
= document.getElementById("course");        var branch =
document.getElementById("branch");        var phone =
document.getElementById("phone");        var email =
document.getElementById("email");        var address =
document.getElementById("address");        if (firstName.value.trim() ===
"") { alert("Please enter your first name.");
        firstName.focus();
return false;
    }
    if (lastName.value.trim() === "") {
alert("Please enter your last name.");
lastName.focus();        return false;
    }
    if (dob.value === "") {
alert("Please enter your date of birth.");
dob.focus();        return false;
    }
}

```

```

        if (!gender) {
alert("Please select your gender.");
return false;
        }
        if (course.value === "") {
alert("Please select your course.");
course.focus();          return false;
        }
        if (branch.value.trim() === "") {
alert("Please enter your branch.");
branch.focus();          return false;
        }
        if (phone.value.trim().length !== 10) {
alert("Phone number should be exactly 10 digits.");
phone.focus();          return false;
        }
        if (!isValidEmail(email.value.trim())) {
alert("Please enter a valid email address.");
email.focus();          return false;
        }
        if (address.value.trim() === "") {
alert("Please enter your address.");
address.focus();          return false;
        }
        return true;
    }
}

```

```

function isValidEmail(email) {          var
emailRegex    =    /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
return emailRegex.test(email); }

```

File Name: 'fl.html'

```

<html>
    <body>
        Successfully submitted the form.<br>
    </body>
</html>

```

Output:-

The screenshot shows a web browser window with the address bar displaying "sers/himan/OneDrive/Desktop/main.html". The browser has two tabs: "webt index.pdf" and "Student Registration Form". A security warning icon is visible in the address bar. A modal dialog box titled "This page says" is overlaid on the form, displaying the message "Please enter a valid email address." with an "OK" button.

STUDENT

First Name:

Last Name:

DATE OF BIRTH:

GENDER: ☒ Male ☐ Female

COURSE:

BRANCH:

PHONE:

EMAIL:

ADDRESS:

PROGRAM – 3

Objective:-

Write programs using Javascript for Web Pages to display browser information.

Program:-

```
<!DOCTYPE html>
<html>
<head>
    <title>Browser Information</title>
</head>
<body>
    <!-- Displaying Browser Information -->
    <h2>Browser Information</h2>
    <p><strong>Browser Name:</strong> <span id="browserName"></span></p>
    <p><strong>Browser Version:</strong> <span id="browserVersion"></span></p>
    <p><strong>Browser Language:</strong> <span id="browserLanguage"></span></p>
    <p><strong>Platform:</strong> <span id="platform"></span></p>
    <p><strong>Cookies Enabled:</strong> <span id="cookiesEnabled"></span></p>
    <p><strong>User Agent:</strong> <span id="userAgent"></span></p>

    <!-- JavaScript to populate browser information -->
    <script>        document.getElementById('browserName').textContent =
navigator.appName;        document.getElementById('browserVersion').textContent =
navigator.appVersion;
        document.getElementById('browserLanguage').textContent =
        navigator.language;
        document.getElementById('platform').textContent = navigator.platform;
document.getElementById('cookiesEnabled').textContent = navigator.cookieEnabled ? 'Yes' :
'No';
        document.getElementById('userAgent').textContent = navigator.userAgent;    </script>
</body>
</html>
```

Output:-

Browser Information

Browser Name: Netscape

Browser Version: 5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/124.0.0.0 Safari/537.36 Edg/124.0.0.0

Browser Language: en-US

Platform: Win32

Cookies Enabled: Yes

User Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/124.0.0.0 Safari/537.36 Edg/124.0.0.0



PROGRAM – 4

Objective:-

Write a Java applet to display the Application Program screen i.e. calculator and other.

Program:-

```
import java.applet.Applet; import
java.awt.Button; import
java.awt.Frame; import
java.awt.Graphics; import
java.awt.TextField; import
java.awt.event.ActionEvent; import
java.awt.event.ActionListener;
import javax.script.ScriptEngine;
import
javax.script.ScriptEngineManager;
import javax.script.ScriptException;
```

```
public class Calculator extends Applet implements ActionListener {      public static
final String[] TEXT = {"AC", "BSp", "%", "/", "7", "8", "9", "*", "4", "5", "6", "-", "1",
"2", "3", "+", "RSET", "0", ".", "="};
```

```
    TextField inputTextField;
    TextField outputTextField;
    StringBuilder inputExpression = new StringBuilder();
```

```
    // Initialize the applet
    public void init() {
        Frame title = (Frame) this.getParent().getParent();

        title.setTitle("Simple Calculator");

        setLayout(null);
```

```
        inputTextField = new TextField("", 4);
        inputTextField.setBounds(50, 10, 200, 50);        add(inputTextField);
```

```

        outputTextField = new TextField("", 4);
outputTextField.setBounds(50, 60, 200, 50);
add(outputTextField);

        int x = 50;
int y = 115;
int k = 1;

        // Create and position buttons        Button[]
buttons = new Button[20];        for (int i = 0; i <
TEXT.length; i++) {        buttons[i] = new
Button("" + TEXT[i]);
buttons[i].setBounds(x * k, y, 50, 50);        if
(k % 4 == 0) {        x = 50;        y
+= 50;        k = 0;        }
k++;        add(buttons[i]);
buttons[i].addActionListener(this);
        }
    }

        // ActionListener implementation

        public void actionPerformed(ActionEvent e) {        String s
= e.getActionCommand();        if (s.equalsIgnoreCase("")) {
inputTextField.setText(calculate(inputExpression));
inputExpression.setLength(0);
        }
        else if (s.equalsIgnoreCase("AC")) {
            inputExpression.setLength(0);
outputTextField.setText("");
        }
        else if (s.equalsIgnoreCase("BSp")) {
            if (inputExpression.length() > 0) {
                inputExpression.deleteCharAt(inputExpression.length() - 1);
outputTextField.setText(inputExpression.toString());
            }
        }
        else if (s.equalsIgnoreCase("RSET")) {
            inputExpression.setLength(0);
inputTextField.setText("");        outputTextField.setText("");
        }
        else {
            inputExpression.append(s);
outputTextField.setText(inputExpression.toString());

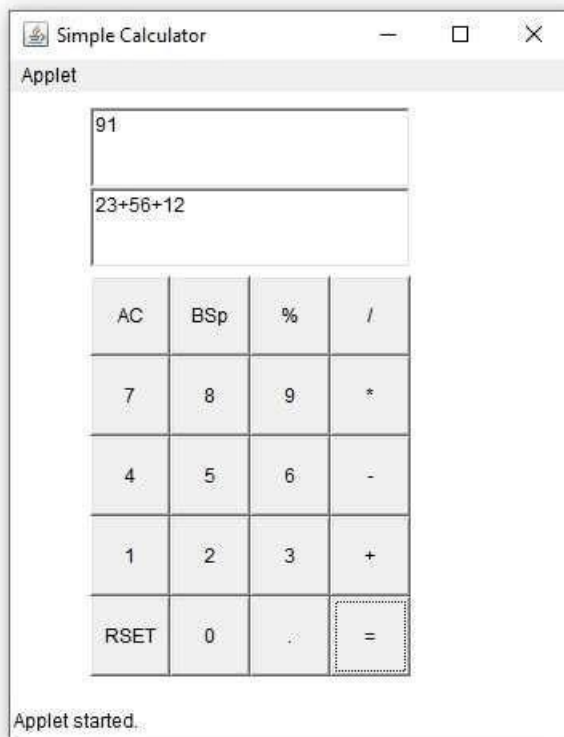
```

```

    }
}
// Method to calculate expression
public String calculate(StringBuilder sb) {
    // Create script engine
    ScriptEngineManager mgr = new ScriptEngineManager();
    ScriptEngine engine = mgr.getEngineByName("JavaScript");
    Object result = null;
    try {
        result = engine.eval(sb.toString());
    } catch (ScriptException ex) {
        result
        = "Error: Invalid expression";
    }
    return result.toString();
}
}

```

Output:-



PROGRAM – 5

Objective:-

Write a program in XML to create DTD, which specifies a set of rules. Create a style sheet in CSS/ XSL & display the document in Internet Explorer.

Program:-

File Name: 'data.xml'

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE products SYSTEM "products.dtd">
<products>
    <product>
        <name>Phone</name>
        <price>500</price>
    </product>
    <product>
        <name>Laptop</name>
        <price>1000</price>
    </product>
    <product>
        <name>Tablet</name>
        <price>300</price>
    </product>
</products>
```

File Name: 'style.css'

```
table {
    border-collapse: collapse;    width: 100%;
}

th, td {
    border: 1px solid #dddddd;    text-align: left;        padding: 8px;
}

th {
```

```
background-color: #f2f2f2;  
}
```

File Name: 'index.html'

```

<!DOCTYPE html>
<html>
<head>
<title>Products</title>
<script>
    function loadXMLDoc(filename) {
        if (window.ActiveXObject) {
            xmlhttp = new
            ActiveXObject("Msxml2.XMLHTTP");
        } else {
            xmlhttp = new XMLHttpRequest();
        }
        xmlhttp.open("GET", filename, false);
        try {
            xmlhttp.responseType = "msxml-document";
        } catch (err) {} // Helping IE11
        xmlhttp.send("");
        return xmlhttp.responseXML;
    }

    function displayXML() {
        xml = loadXMLDoc("data.xml");
        xsl = loadXMLDoc("style.xsl");
        if (window.ActiveXObject || xmlhttp.responseType=="msxml-
        document") {
            ex = xml.transformNode(xsl);
            document.getElementById("example").innerHTML = ex;
        } else if
        (document.implementation &&
            document.implementation.createDocument) {
            xsltProcessor = new XSLTProcessor();
            xsltProcessor.importStylesheet(xsl);
            resultDocument = xsltProcessor.transformToFragment(xml, document);
            document.getElementById("example").appendChild(resultDocument);
        }
    }
</script>
</head>
<body onload="displayXML()">
    <div id="example"></div>
</body>
</html>

```

File Name: 'products.dtd'


```
<!ELEMENT products (product+)>
<!ELEMENT product (name, price)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT price (#PCDATA)>
```

File Name: 'style.xml'

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="style.css" />
  </head>
  <body>
    <h2>Products List</h2>
    <table>
      <tr>
        <th>Name</th>
        <th>Price</th>
      </tr>
      <xsl:for-each select="products/product">
        <tr>
          <td><xsl:value-of select="name"/></td>
          <td><xsl:value-of select="price"/></td>
        </tr>
      </xsl:for-each>
    </table>
  </body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Output:-



The screenshot shows a web browser window with a single tab titled 'Products'. The address bar displays the URL '127.0.0.1:3000/WT%20Lab/index.html'. The main content area features a table titled 'Products List'.

Name	Price
Phone	500
Laptop	1000
Tablet	300

PROGRAM – 6

Objective:-

Program to illustrate JDBC connectivity. Program for maintaining the database by sending queries. Design and implement a simple servlet book query with the help of JDBC & SQL. Create MS Access Database, Create on ODBC link, Compile & execute JAVA JDBC.

Program:-

Step 1: Create a SQL database.

CREATE DATABASE library; USE library;

CREATE TABLE books (id INT PRIMARY KEY
AUTO_INCREMENT, title VARCHAR(255), author
VARCHAR(255), price DECIMAL(10, 2), quantity INT
);

Step 2: Write JDBC Connectivity Program.

```
import java.sql.*; public class JDBCExample {      private static final String
JDBC_URL = "jdbc:mysql://localhost:3306/library";      private static final
String USERNAME = "your_username";      private static final String
PASSWORD = "your_password";      public static void main(String[] args) {
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection connection = DriverManager.getConnection(JDBC_URL,
            USERNAME, PASSWORD);
        Statement statement = connection.createStatement();
        ResultSet resultSet = statement.executeQuery("SELECT * FROM books");
        while (resultSet.next()) {
            int id = resultSet.getInt("id");
            String title = resultSet.getString("title");      String author
= resultSet.getString("author");      int publishedYear =
resultSet.getInt("published_year");
            System.out.println("Book ID: " + id);
            System.out.println("Title: " + title);
            System.out.println("Author: " + author);
            System.out.println("Published Year: " + publishedYear);
            System.out.println();
        }
        resultSet.close();
        statement.close();      connection.close();
    } catch (ClassNotFoundException | SQLException e) {
        e.printStackTrace();
    }
}
```

```

    }
}

```

Step 3: Write servlet for Book Query.

```

import java.io.*; import javax.servlet.*; import javax.servlet.http.*; import java.sql.*;
public class BookServlet extends HttpServlet {    private static final String
JDBC_URL = "jdbc:mysql://localhost:3306/library";    private static final String
USERNAME = "your_username";    private static final String PASSWORD =
"your_password";    protected void doGet(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection connection = DriverManager.getConnection(JDBC_URL,
            USERNAME, PASSWORD);
        Statement statement = connection.createStatement();
        ResultSet resultSet = statement.executeQuery("SELECT * FROM books");
        out.println("<html><body>");
        out.println("<h2>Books</h2>");
        out.println("<table border='1'><tr><th>ID</th> <th>Title</th>
            <th>Author</th> <th>Published Year</th></tr>");
        while (resultSet.next()) {
            int id = resultSet.getInt("id");
            String title = resultSet.getString("title");
            String author =
resultSet.getString("author");
            int publishedYear =
resultSet.getInt("published_year");
            out.println("<tr><td>" + id +
"</td><td>" + title + "</td><td>" + author + "</td><td>" + publishedYear +
"</td></tr>");
        }
        out.println("</table>");
        out.println("</body></html>");
        resultSet.close();
        statement.close();
        connection.close();
    } catch (ClassNotFoundException | SQLException e) {
        e.printStackTrace();
    }
}
}

```

PROGRAM – 7

Objective:-

Install the TOMCAT web server and APACHE. Access the above-developed static web pages for the book website, using these server by putting the web pages developed.

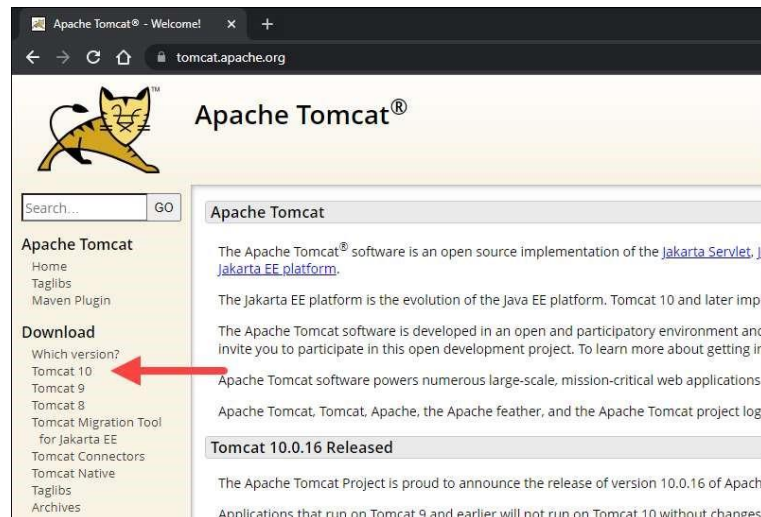
Introduction:-

Apache Tomcat is an open-source web server and servlet container for Java code. Tomcat executes programs written in the Java programming language and implements many Java EE specifications, including Jakarta Servlet, Jakarta Server Pages, and others.

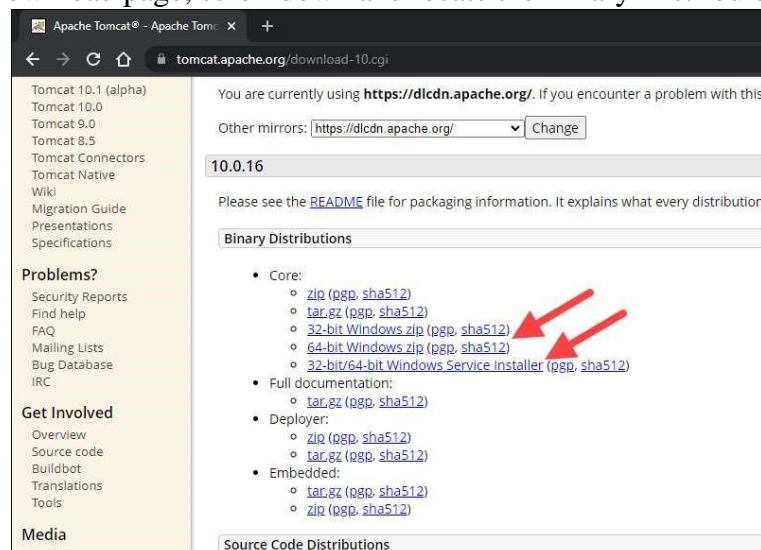
How to Install Tomcat on Windows:

Step 1: Download Tomcat for Windows.

- i. Browse to the official Apache Tomcat website. Locate the Download section and click the latest Tomcat version available.



- ii. On the Download page, scroll down and locate the Binary Distributions area.

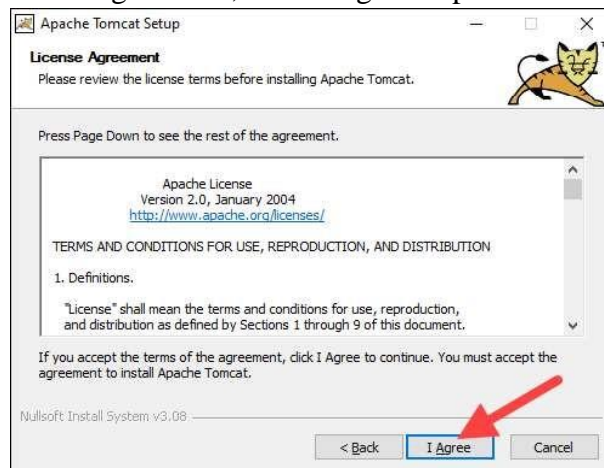


Step 2: Follow the steps below to install Tomcat using the Windows Service Installer.

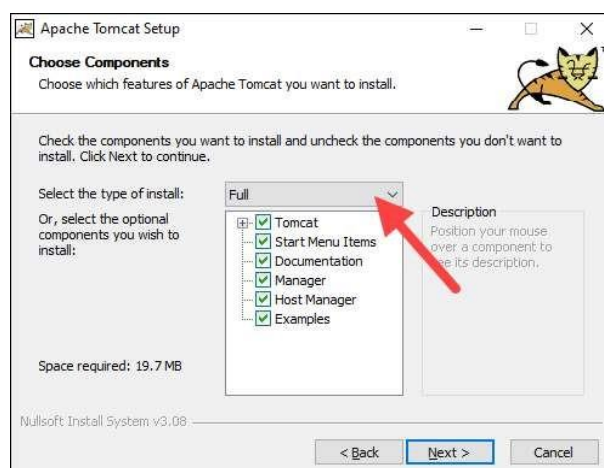
- i. Open the Windows Service Installer file to start the installation process.
- ii. In the Tomcat Setup welcome screen, click Next to proceed.



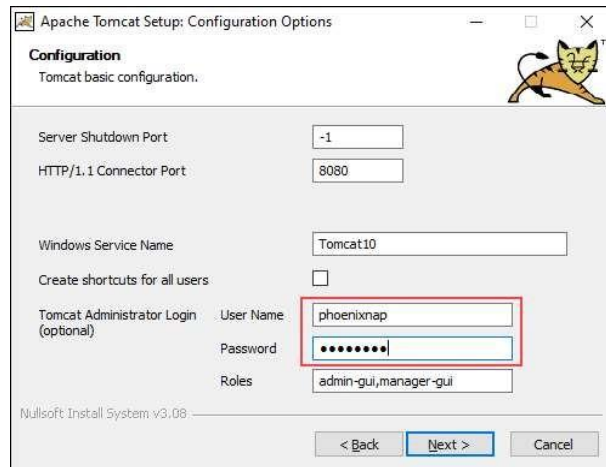
- iii. Read the License Agreement, click I Agree to proceed to the next step.



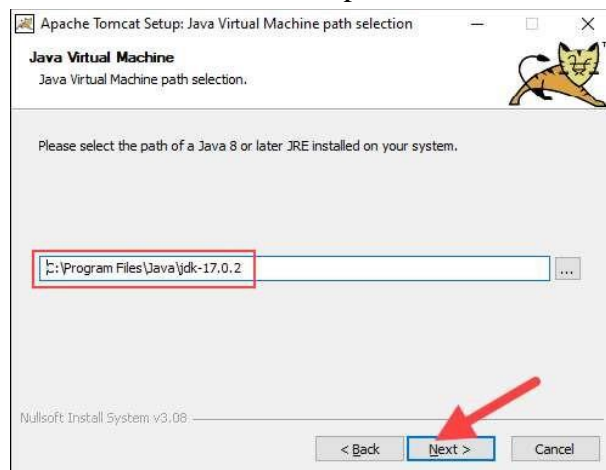
- iv. In the Tomcat component select on screen, choose Full in the dropdown menu to ensure the wizard installs the Tomcat Host Manager and Servlet & JSP examples web applications. Alternatively, keep the default Normal installed on the type and click Next.



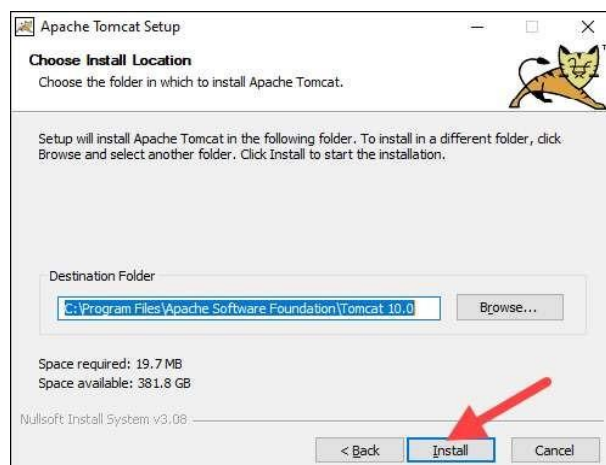
- v. The next step is configuring the Tomcat server. For instance, enter the Administrator login credentials or choose a different connect the port. When finished, click Next to proceed to the next step.



- vi. The next step requires you to enter the full path to the JRE directory on your system. The wizard auto-completes this if you have previously set up the Java environment variables. Click Next to proceed to the next step.

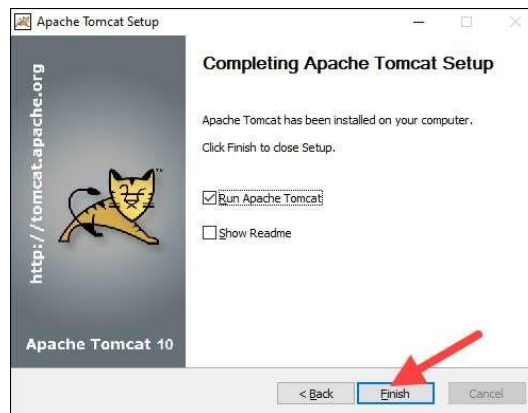


- vii. Choose the Tomcat server install location or keep the default one and click Install.

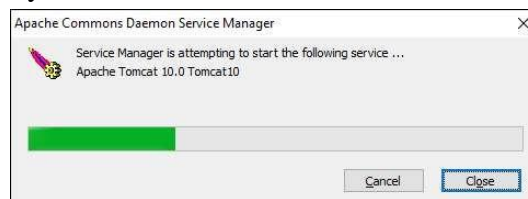


- viii. Check the Run Apache Tomcat box to start the service after the installation finishes.

Optionally, check the Show Readme box to see the Readme file. To complete the installation, click Finish.



- ix. A popup window appears that starts the Tomcat service. After the process completes, the window closes automatically. The Apache Tomcat web server is now successfully installed.



Step 3: Check if the Apache Tomcat Service Is Running.

Installing Tomcat using the Windows Service Installer installs Tomcat as a Windows service that automatically runs on boot.

PROGRAM – 8

Objective:-

Install a database (Mysql or Oracle). Create a table which should contain at least the following fields: name, password, email-id, phone number Write a Java program/servlet/JSP to connect to that database and extract data from the tables and display them. Insert the details of the users who register with the website, whenever a new user clicks the submit button on the registration page.

Program:-

Step 1: Database Installation on (MySQL)

- i. Download and install MySQL from the official website according to your operating system.
- ii. Once installed, start the MySQL server and open a terminal or command prompt.
- iii. Log in to MySQL using the command `mysql -u username -p`, where username is your MySQL username.
- iv. Create a new database for our example, e.g., `CREATE DATABASE mywebsite_db;`
- v. Switch to the newly created database: `USE mywebsite_db;`
- vi. Create a table named users with the required fields:

```
CREATE TABLE users ( id INT
    AUTO_INCREMENT PRIMARY
    KEY, name VARCHAR(255)
    NOT NULL, password
    VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT
    NULL, phone VARCHAR(15)
    NOT NULL
);
```

Step 2: JSP Code for Database Connectivity and User Registration

- i. Create a new JSP file in your web application directory, e.g., register.jsp.
- ii. Add the following code to establish a connection to the MySQL database and insert user registration details:

File name: 'web.xml'

```
<web-app>
    <servlet>
        <servlet-name>init1</servlet-name>
        <servlet-class>Ini</servlet-class>
    </servlet>
    <servlet-mapping>
```

```
        <servlet-name>init1</servlet-name>
        <url-pattern>/regis</url-pattern>
    </servlet-mapping>
</web-app>
File name: 'registra on.html'
<html>
<head>
    <title>Registration page</title>
</head>
<body bgcolor="#00FFFF">
    <form method="POST" action="register">
        <center>
            <table>
                <tr>
                    <td>Username</td>
                    <td><input type="text" name="usr"></td>
                </tr>
                <tr>
                    <td>Password</td>
                    <td><input type="password" name="pwd"></td>
                </tr>
                <tr>
                    <td>Age</td>
                    <td><input type="text" name="age"></td>
                </tr>
                <tr>
                    <td>Address</td>
                    <td><input type="text" name="add"></td>
                </tr>
                <tr>
                    <td>Email</td>
                    <td><input type="text" name="mail"></td>
                </tr>
                <tr>
                    <td>Phone</td>
                    <td><input type="text" name="phone"></td>
                </tr>
                <tr>
                    <td colspan="2" align="center"><input
                        type="submit" value="Submit"></td>
                </tr>
            </table>
        </center>
    </form>
</body>
</html>
```

```
        </tr>
    </table>
</center>
</form>
</body>
</html>
```

Output:-

Username	<input type="text"/>
Password	<input type="password"/>
Age	<input type="text"/>
Address	<input type="text"/>
email	<input type="text"/>
Phone	<input type="text"/>
<input type="submit" value="submit"/>	

PROGRAM – 9

Objective:-

Design and implement a simple shopping cart example with session tracking API.

Program:-

File name:

‘shoppingCart.java’

import java.io.*; import

javax.servlet.*; import

javax.servlet.http.*;

```
public class ShoppingCart extends HttpServlet {
    public void service(HttpServletRequest req, HttpServletResponse res) throws
        ServletException, IOException {
        String str1 = req.getParameter("item");
        String str2 = req.getParameter("qty");
        String str3 = req.getParameter("add");
        String str4 = req.getParameter("list");

        res.setContentType("text/html");
        PrintWriter out = res.getWriter();

        if (str3 != null) {
            Cookie c1 = new Cookie(str1, str2);
            res.addCookie(c1);          res.sendRedirect("ShoppingCart.html");
        } else if (str4 != null) {
            Cookie clientCookies[] = req.getCookies();
            for (int i = 0; i < clientCookies.length; i++) {
                out.print("<B>" + clientCookies[i].getName() + " : " +
                    clientCookies[i].getValue() + "</B><BR>");
            }
        }
        out.close();
    }
}
```

File name: ‘web.html’

<servlet>

<servlet-name>snrao1</servlet-name>

<servlet-class>ShoppingCart</servlet-class>

```
</servlet>
<servlet-mapping>
    <servlet-name>snrao1</servlet-name>
    <url-pattern>/SC</url-pattern>
</servlet-mapping>
```

File name: 'shoppingCart.html'

```
<html>
<head>
    <title>Cookie Example through Shopping Cart</title>
</head>
<body>
    <h3>Cookie Example through Shopping Cart</h3>
    <form method="get" action="http://localhost:8888/india/SC">
        Enter Item Name <input type="text" name="item"><br>
        Enter Item Quantity <input type="text" name="qty"><br>
        <input type="submit" value="Add Cookie" name="add">
        <input type="submit" value="List Cookies" name="list">
    </form>
</body>
</html>
```

Output:-

Cookie Example through Shopping Cart

Enter Item Name

Enter Item Quantity

Add Cookie

List Cookies

PROGRAM – 10

Objective:-

Create an html page named “Table.html” to display your class time table. (i) Provide the title as “Time Table”. (ii) Provide various color options to the cells like lab hours and elective hours (with different color options).

Program:-

```
<html>
<head>
<title>My Time Table</title>
<style>
td,
th {
font-family: Verdana, sans-serif;
color: black;
text-align: center;
font-weight: bold;
border: 2px solid black;
padding: 5px;
font-style: italic;
}

th {
font-weight: bolder;
}
h2 {
text-align: center;
font-style: italic;
}

table {
border-collapse: collapse;
width: 100%;
margin-bottom: 20px;
border: 2px solid black;
}

.lunch td {
font-family: Verdana, sans-serif;
```

```

        color: black;
        writing-mode: vertical-lr;
        font-weight: bold;
        text-align: center;
        padding: 5px;
    }
</style>
</head>
<body>
    <h2>
        <b>
            ><i><u>TIME-TABLE SESSION 2023-2024</u></i></b>
        >
    </h2>
    <h2>
        <b>
            ><i><u>(SECTION - 3C)</u></i></b>
        >
    </h2>
    <table>
        <tr>
            <th colspan="2" rowspan="2" bgcolor="BFCFE7">DATE/TIME</th>
            <th colspan="2" rowspan="2" bgcolor="BFCFE7">09:20AM - 10:10AM</th>
            <th colspan="2" rowspan="2" bgcolor="BFCFE7">10:11AM - 11:00AM</th>
            <th colspan="2" rowspan="2" bgcolor="BFCFE7">11:01AM - 11:50AM</th>
            <th colspan="2" rowspan="2" bgcolor="BFCFE7">11:51AM - 12:50PM</th>
            <th colspan="2" rowspan="2" bgcolor="BFCFE7">12:51PM - 01:30PM</th>
            <th colspan="2" rowspan="2" bgcolor="BFCFE7">01:31PM - 02:20PM</th>
            <th colspan="2" rowspan="2" bgcolor="BFCFE7">02:21PM - 03:10PM</th>
            <th colspan="2" rowspan="2" bgcolor="BFCFE7">03:11PM - 04:00PM</th>
            <th colspan="2" rowspan="2" bgcolor="BFCFE7">04:01PM - 04:50PM</th>
        </tr>
        <tr></tr>
        <tr>
            <td colspan="2" rowspan="2" bgcolor="535C91">MONDAY</td>
            <td colspan="4" rowspan="2" bgcolor="#F5DD61">Technical Training</td>
            <td colspan="4" rowspan="1" bgcolor="50623A">SE LAB (G1)</td>
            <td colspan="2" rowspan="10" bgcolor="FDDD9F">LUNCH</td>
            <td colspan="2" rowspan="2" bgcolor="#EE4266">SPM</td>
        </tr>

```

WT		Big Data		Sports	
WT (G2)		CN (G2)			
TUESDAY		Aptitude and Verbal		SE	
ITCS		WT Lab (G2)		Big Data	
CN		CN Lab (G1)			
WEDNESDAY		Technical Training		WT	
SPM		Big Data		SE Tutorial (G2)	
EXTRA CURRICULAR ACTIVITIES					
CN Tutorial (G1)					
THURSDAY		Aptitude and Verbal		SE	
SPM					


```

        <td colspan="4" rowspan="1" bgcolor="EE99C2">WT Lab (G1)</td>
        <td colspan="2" rowspan="2" bgcolor="9BCF53">CN</td>
        <td colspan="2" rowspan="2" bgcolor="#0096FF">ITCS</td>
    </tr>
    <tr>
        <td colspan="4" rowspan="1" bgcolor="50623A">SE LAB (G2)</td>
    </tr>
    <tr>
        <td colspan="2" rowspan="2" bgcolor="535C91">FRIDAY</td>
        <td colspan="4" rowspan="2" bgcolor="#F5DD61">Technical Training</td>
        <td colspan="4" rowspan="1" bgcolor="74E291">CN Lab (G1)</td>
        <td colspan="2" rowspan="2" bgcolor="green">SE</td>
        <td colspan="2" rowspan="2" bgcolor="#FF004D">WT</td>
        <td colspan="2" rowspan="2" bgcolor="9BCF53">CN</td>
        <td colspan="2" rowspan="2" bgcolor="yellow">LIB</td>
    </tr>
    <tr>
        <td colspan="2" rowspan="1" bgcolor="FB88B4">WT Tutorial (G2)</td>
        <td colspan="2" rowspan="1" bgcolor="86A789">SE Tutorial (G1)</td>
    </tr>
    <tr>
        <td colspan="2" rowspan="2" bgcolor="535C91">SATURDAY</td>
        <td colspan="2" rowspan="2" bgcolor="FFD0EC"></td>
        <td colspan="2" rowspan="2" bgcolor="FFD0EC"></td>
        <td colspan="2" rowspan="2" bgcolor="FFD0EC"></td>
        <td colspan="2" rowspan="2" bgcolor="FFD0EC"></td>
        <td colspan="2" rowspan="2" bgcolor="FFD0EC"></td>
        <td colspan="2" rowspan="2" bgcolor="FFD0EC"></td>
        <td colspan="2" rowspan="2" bgcolor="FFD0EC"></td>
    </tr>
    </table>
</body>
</html>

```

Output:-

My Time Table >

(SECTION - 3C)

DATE/TIME	09:20AM - 10:10AM	10:11AM - 11:00AM	11:01AM - 11:50AM	11:51AM - 12:50PM	12:51PM - 01:30PM	01:31PM - 02:20PM	02:21PM - 03:10PM	03:11PM - 04:00PM	04:01PM - 04:50PM	
MONDAY	Technical Training	SE LAB (G1)		LUNCH	SPM	WT	Big Data	Sports		
		WT (G2)	CN (G2)							
TUESDAY	Aptitude and Verbal	SE	ITCS		WT Lab (G2)		Big Data	CN		
					CN Lab (G1)					
WEDNESDAY	Technical Training	WT	SPM		Big Data	SE Tutorial (G2)	EXTRA CURRICULAR ACTIVITIES			
						CN Tutorial (G1)				
THURSDAY	Aptitude and Verbal	SE	SPM		WT Lab (G1)		CN	ITCS		
					SE LAB (G2)					
FRIDAY	Technical Training	CN Lab (G1)				SE	WT	CN	LIB	
		WT Tutorial (G2)	SE Tutorial (G1)							
SATURDAY										

44

PROGRAM – 11

Objective:-

Write HTML code to develop a web page having frames that divide the page into two equal rows and divide second column into two columns (30% and 70%). Column-1 contain html page contains below link. (i) Personal Information (ii) Qualification (iii) Hobbies When user click on any link, related page should be open in Column-2.

Program:-

Index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Frames Example</title>
</head>
<frameset rows="50%, 50%" frameborder="yes" border="1">
    <!-- First Row -->
    <frame src="content.html" name="content_frame">

    <!-- Second Row Split into Two Columns -->
    <frameset cols="30%, 70%">
        <!-- Links in the first column -->
        <frame src="links.html" name="links_frame">
        <!-- Content display area in the second column -->
        <frame src="about.html" name="display_frame">
    </frameset>
</frameset>

<noframes>
    <body>
        Your browser does not support frames.
    </body>
</noframes>
</html>
```

Personal_info.html

```
<!DOCTYPE html>
<html>
```

```
<head>
  <title>Personal Information</title>
</head>
<body>
  <h1>Personal Information</h1>
  <p>Name: Himanshu Pathank</p>
  <p>Email: himanshupathak543@gmail.com</p>
  <p>Address: Greater Noida</p>
</body>
</html>
```

Content.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Welcome Content</title>
  </head>
  <body>
    <h1>Welcome to My Page</h1>
    <p>
      This is the main content area in the first frame. It can include any
      general information, introduction, or overview of the website.
    </p>
  </body>
</html>
```

Links.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Links Page</title>
  </head>
  <body>
    <p>
      <a href="personal_info.html" target="display_frame">
        >Personal Information</a>
    >
  </body>
</html>
```

```
</p>
<p><a href="qualification.html" target="display_frame">Qualification</a></p>
<p><a href="hobbies.html" target="display_frame">Hobbies</a></p>
</body>
</html>
```

Qualifications.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Qualification</title>
  </head>
  <body>
    <h1>Qualification</h1>
    <p>
      Bachelor of Science in Computer Science from GL BAJAJ INSTITUTE OF
      TECHNOLOGY AND MANAGEMENT
    </p>
  </body>
</html>
```

Hobbies.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hobbies</title>
  </head>
  <body>
    <h1>Hobbies</h1>
    <ul>
      <li>Reading</li>
      <li>Photography</li>
      <li>Hiking</li>
      <li>Coding</li>
    </ul>
  </body>
</html>
```

Output:-

Welcome to My Page <small>This is the main content area in the first frame. It can include any general information, introduction, or overview of the website.</small>	
Personal Information Qualification Hobbies	Personal Information Name: Himanshu Pathank Email: himanshupathak543@gmail.com Address: Greater Noida

Welcome to My Page <small>This is the main content area in the first frame. It can include any general information, introduction, or overview of the website.</small>	
Personal Information Qualification Hobbies	Qualification Bachelor of Science in Computer Science from GL BAJAJ INSTITUTE OF TECHNOLOGY AND MANAGEMENT

Welcome to My Page <small>This is the main content area in the first frame. It can include any general information, introduction, or overview of the website.</small>	
Personal Information Qualification Hobbies	Hobbies <ul style="list-style-type: none">• Reading• Photography• Hiking• Coding

PROGRAM – 12

Objective:-

Write a program in java to demonstrate the constructor overloading.

Program:-

```
public class Rectangle {
    // Instance variables
    private double length;
    private double width;

    // Default constructor
    public Rectangle() {
        this.length = 1;
        this.width = 1;
        System.out.println("Constructed a rectangle with default size.");
    }

    // Constructor with one double parameter
    public Rectangle(double size) {
        this.length = size;
        this.width = size;
        System.out.println("Constructed a square with size " + size);
    }

    // Constructor with two double parameters
    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
        System.out.println("Constructed a rectangle with length " + length + " and width " +
width);
    }

    // Method to compute area of the rectangle
    public double getArea() {
        return length * width;
    }

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

```
// Create objects using different constructors
Rectangle defaultRectangle = new Rectangle();
Rectangle square = new Rectangle(5);
Rectangle rectangle = new Rectangle(4, 5);

// Print areas of the objects
System.out.println("Area of default rectangle: " + defaultRectangle.getArea());
System.out.println("Area of square: " + square.getArea());
System.out.println("Area of rectangle: " + rectangle.getArea());
}
}
```

Output:-

Output

```
Constructed a rectangle with default size.
Constructed a square with size 5.0
Constructed a rectangle with length 4.0 and width 5.0
Area of default rectangle: 1.0
Area of square: 25.0
Area of rectangle: 20.0
```


PROGRAM – 13

Objective:-

i) Write a program to Implement Single and multilevel inheritance using Java.

Program:-

// Single Inheritance Example: Class Animal is the parent class

```
class Animal {  
    void eat() {  
        System.out.println("This animal eats food.");  
    }  
}
```

// Dog inherits from Animal (Single Inheritance)

```
class Dog extends Animal {  
    void bark() {  
        System.out.println("The dog barks.");  
    }  
}
```

// Multilevel Inheritance Example: Beagle extends Dog, which extends Animal

```
class Beagle extends Dog {  
    void sniff() {  
        System.out.println("The beagle sniffs.");  
    }  
}
```

```
public class InheritanceExample {  
    public static void main(String[] args) {  
        Beagle myBeagle = new Beagle();  
        myBeagle.eat(); // From Animal class  
        myBeagle.bark(); // From Dog class  
        myBeagle.sniff(); // From Beagle class  
    }  
}
```

Output:-

Output

```
This animal eats food.  
The dog barks.  
The beagle sniffs.
```

ii) Write a program to Implement multiple inheritance. [note : use Interface]

Program:-

```
interface WaterAnimal {
    void swim();
}

interface LandAnimal {
    void walk();
}

// Multiple inheritance using interfaces
class Crocodile implements WaterAnimal, LandAnimal {
    public void swim() {
        System.out.println("The crocodile swims.");
    }

    public void walk() {
        System.out.println("The crocodile walks on land.");
    }
}

public class InterfaceExample {
    public static void main(String[] args) {
        Crocodile myCrocodile = new Crocodile();
        myCrocodile.swim(); // Implements method from WaterAnimal
        myCrocodile.walk(); // Implements method from LandAnimal
    }
}
```

Output

```
The crocodile swims.
The crocodile walks on land.
```

PROGRAM – 14

Objective:-

Write a program to Implement exception handling using Java.

Program:-

```
public class ExceptionHandlingExample {  
    public static void main(String[] args) {  
        int numerator = 10;  
        int denominator = 0; // This will cause an ArithmeticException  
        try {  
            int result = numerator / denominator;  
            System.out.println("The result is " + result);  
        } catch (ArithmeticException e) {  
            System.out.println("Error: Cannot divide by zero.");  
        } finally {  
            System.out.println("This block is executed regardless of the exception occurrence.");  
        }  
  
        System.out.println("Program continues after exception handling.");  
    }  
}
```

Output

```
Error: Cannot divide by zero.  
This block is executed regardless of the exception occurrence.  
Program continues after exception handling.
```

PROGRAM – 15

Objective:-

Write a program to implement concept of multithreading using Java.

Program:-

Using the Thread class

```
class CountThread extends Thread {
    private String threadName;

    CountThread(String name) {
        this.threadName = name;
    }

    public void run() {
        for (int i = 1; i <= 5; i++) {
            System.out.println(threadName + ": " + i);
            try {
                Thread.sleep(1000); // Thread sleeps for 1 second
            } catch (InterruptedException e) {
                System.out.println(threadName + " interrupted.");
                // Re-interrupt the thread to handle it properly outside
                Thread.currentThread().interrupt();
            }
        }
        System.out.println(threadName + " has finished executing.");
    }

    public static void main(String[] args) {
        CountThread thread1 = new CountThread("Thread 1");
        CountThread thread2 = new CountThread("Thread 2");

        thread1.start();
        thread2.start();
    }
}
```

Method 2: Using the Runnable interface

```
class CountRunnable implements Runnable {
    private String threadName;

    CountRunnable(String name) {
        this.threadName = name;
    }

    @Override
    public void run() {
        for (int i = 1; i <= 5; i++) {
            System.out.println(threadName + ": " + i);
            try {
                Thread.sleep(1000); // Thread sleeps for 1 second
            } catch (InterruptedException e) {
                System.out.println(threadName + " interrupted.");
                // Re-interrupt the thread to handle it properly outside
                Thread.currentThread().interrupt();
            }
        }
        System.out.println(threadName + " has finished executing.");
    }

    public static void main(String[] args) {
        Thread thread1 = new Thread(new CountRunnable("Thread 1"));
        Thread thread2 = new Thread(new CountRunnable("Thread 2"));

        thread1.start();
        thread2.start();
    }
}
```

Output:-

```
Output Clear  
java -cp /tmp/zTtYvFK6yUE/CountThread  
Thread 1: 1  
Thread 2: 1  
Thread 1: 2  
Thread 2: 2  
Thread 1: 3  
Thread 2: 3  
Thread 1: 4  
Thread 2: 4  
Thread 1: 5  
Thread 2: 5  
Thread 1 has finished executing.  
Thread 2 has finished executing.  
  
=== Code Execution Successful ===
```

```
Output Clear  
java -cp /tmp/AdVpit8AA2/CountRunnable  
Thread 2: 1  
Thread 1: 1  
Thread 2: 2  
Thread 1: 2  
Thread 2: 3  
Thread 1: 3  
Thread 2: 4  
Thread 1: 4  
Thread 2: 5  
Thread 1: 5  
Thread 1 has finished executing.  
Thread 2 has finished executing.  
  
=== Code Execution Successful ===
```

PROGRAM – 16

Objective:-

Write a script for name validation, mobile number validation and email validation

Program:-

Name Validation Script

```
import java.util.regex.*;

public class NameValidation {
    public static boolean validateName(String name) {
        // Regex pattern for validating name (allowing alphabets and spaces, at least 2 characters)
        String regex = "^[a-zA-Z]+(\\s[a-zA-Z]+)?$";
        Pattern pattern = Pattern.compile(regex);
        Matcher matcher = pattern.matcher(name);
        return matcher.matches();
    }

    public static void main(String[] args) {
        // Test name validation
        String[] names = {"John Doe", "Alice", "123", "John123", "Bob!"};
        for (String name : names) {
            System.out.println(name + " is valid: " + validateName(name));
        }
    }
}
```

Mobile Number Validation Script

```
import java.util.regex.*;

public class MobileNumberValidation {
    public static boolean validateMobileNumber(String number) {
        // Regex pattern for validating mobile number (optional country code +91 or 91, followed by 10 digits)
        String regex = "^(?:\\+?91)?[7-9][0-9]{9}$";
        Pattern pattern = Pattern.compile(regex);
        Matcher matcher = pattern.matcher(number);
    }
}
```



```

        return matcher.matches();
    }

    public static void main(String[] args) {
        // Test mobile number validation
        String[] numbers = {"+919876543210", "9876543210", "0987654321",
"+91876543210", "876543210"};
        for (String number : numbers) {
            System.out.println(number + " is valid: " + validateMobileNumber(number));
        }
    }
}

```

Email Validation Script

```

import java.util.regex.*;

public class EmailValidation {
    public static boolean validateEmail(String email) {
        // Regex pattern for validating email
        String regex = "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}$";
        Pattern pattern = Pattern.compile(regex);
        Matcher matcher = pattern.matcher(email);
        return matcher.matches();
    }

    public static void main(String[] args) {
        // Test email validation
        String[] emails = {"user@example.com", "user.name@example.co.in",
"user.name@sub.domain.com", "user@name", "user@domain.com!"};
        for (String email : emails) {
            System.out.println(email + " is valid: " + validateEmail(email));
        }
    }
}

```

Output:-

Output

```
|John Doe is valid: true  
Alice is valid: true  
123 is valid: false  
John123 is valid: false  
Bob! is valid: false
```

Output

```
|+919876543210 is valid: true  
9876543210 is valid: true  
0987654321 is valid: false  
+91876543210 is valid: false  
876543210 is valid: false
```

Output

```
|user@example.com is valid: true  
user.name@example.co.in is valid: true  
user.name@sub.domain.com is valid: true  
user@name is valid: false  
user@domain.com! is valid: false
```

PROGRAM – 17

Objective:-

Write a script for login ID page validation. Assume no field can be blank and the password should not be weak.

Program:-

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Login Page Validation</title>

    <script>

        function validateForm() {

            const username = document.getElementById('username').value.trim();

            const password = document.getElementById('password').value.trim();

            // Check if any field is blank

            if (!username || !password) {

                alert('Neither username nor password can be blank.');
```

return false;

```
            }

            // Check password strength

            if (!isStrongPassword(password)) {

                alert('Password is too weak. It must be at least 8 characters long, include uppercase and lowercase letters, a number, and a special character.');
```

return false;

```
            }

            alert('Form submitted successfully!');

            return true;
```

```
}
```

```
function isStrongPassword(password) {
```

```
    const strongPasswordRegex = /^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*?&])[A-Za-z\d@$!%*?&]{8,}$/;
```

```
    return strongPasswordRegex.test(password);
```

```
}
```

```
</script>
```

```
</head>
```

```
<body>
```

```
<h2>Login Form</h2>
```

```
<form onsubmit="return validateForm()">
```

```
<label for="username">Username:</label><br>
```

```
<input type="text" id="username" name="username"><br><br>
```

```
<label for="password">Password:</label><br>
```

```
<input type="password" id="password" name="password"><br><br>
```

```
<input type="submit" value="Submit">
```

```
</form>
```

```
</body>
```

```
</html>
```

Output:-

A screenshot of a web browser window. The address bar shows several tabs: Gmail, YouTube, Maps, News, Translate, and Question Paper Ban... The main content area is titled "Login Form". It contains two input fields: "Username:" with the text "admin" and "Password:" with masked characters "*****". Below these is a "Submit" button. A light blue notification box in the top right corner displays the message "127.0.0.1:5500 says Form submitted successfully!" with an "OK" button.

Login Form

Username:
admin

Password:

Submit

127.0.0.1:5500 says
Form submitted successfully!

OK

A screenshot of a web browser window, similar to the one above. The address bar shows the same tabs. The "Login Form" is displayed with empty "Username:" and "Password:" fields and a "Submit" button. A light blue notification box in the top right corner displays the message "127.0.0.1:5500 says Neither username nor password can be blank." with an "OK" button.

Login Form

Username:

Password:

Submit

127.0.0.1:5500 says
Neither username nor password can be blank.

OK

PROGRAM – 18

Objective:-

Design a JavaScript program to display the current day and time in the following format. Today is : Monday. Current time is : 11 AM : 50 : 58.

Program:-

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Current Day and Time</title>

</head>

<body>

  <h1>Current Day and Time</h1>

  <p id="dayTimeDisplay">Loading time...</p>

  <script>

    function displayCurrentDayAndTime() {

      // Create a new Date object to get the current date and time

      const now = new Date();

      // Days array to convert day index to name

      const days = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"];

      // Get the current day of the week

      const dayOfWeek = days[now.getDay()];

      // Get current hour, minute and second

      let hours = now.getHours();
```

```
const minutes = now.getMinutes();
const seconds = now.getSeconds();

// Determine AM or PM suffix
const ampm = hours >= 12 ? 'PM' : 'AM';

// Convert 24-hour time to 12-hour time
hours = hours % 12;
hours = hours ? hours : 12; // the hour '0' should be '12'

// Format minutes and seconds to always be two digits
const formattedMinutes = minutes < 10 ? '0' + minutes : minutes;
const formattedSeconds = seconds < 10 ? '0' + seconds : seconds;

// Create a formatted string
const formattedTime = `${hours} ${ampm} : ${formattedMinutes} :
${formattedSeconds}`;

// Display the current day and time
document.getElementById('dayTimeDisplay').textContent = `Today is :
${dayOfWeek}. Current time is : ${formattedTime}.`;
}

// Call the function to display the day and time
displayCurrentDayAndTime();

// Update the display every second
setInterval(displayCurrentDayAndTime, 1000);
</script>
</body>
</html>
```

Output:-

Current Day and Time

Today is : Wednesday. Current time is : 7 PM : 27 : 52.

PROGRAM – 19

Objective:-

Develop a simple JSP application that accepts the user's age and displays a message according to the following rules:

- i) Age < 15: Message: You are a Kid!
- (ii) Age between 16 and 40: You are young!
- (iii) Age above 40: You are old!

Program:-

Index.jsp

```
<!DOCTYPE html>

<html>

<head>

    <title>Enter Your Age</title>

</head>

<body>

    <h1>Age Input</h1>

    <form action="processAge.jsp" method="post">

        <label for="age">Enter your age:</label>

        <input type="text" id="age" name="age" required>

        <button type="submit">Submit</button>

    </form>

</body>

</html>
```

processAge.jsp

```
<% @ page import="java.lang.Integer" %>

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

<html>

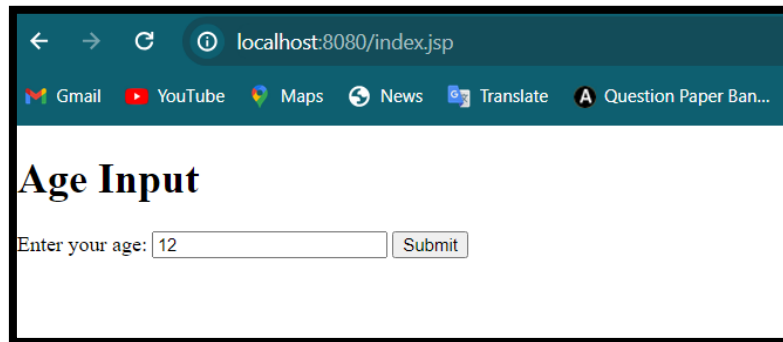
<head>

    <title>Age Response</title>

</head>
```

```
<body>
  <h1>Result</h1>
  <%
    String ageStr = request.getParameter("age");
    if (ageStr != null) {
      int age = Integer.parseInt(ageStr);
      if (age < 15) {
        out.println("<p>You are a Kid!</p>");
      } else if (age >= 16 && age <= 40) {
        out.println("<p>You are young!</p>");
      } else {
        out.println("<p>You are old!</p>");
      }
    } else {
      out.println("<p>Age is required!</p>");
    }
  %>
  <a href="index.jsp">Try Again</a>
</body>
</html>
```

Output:-

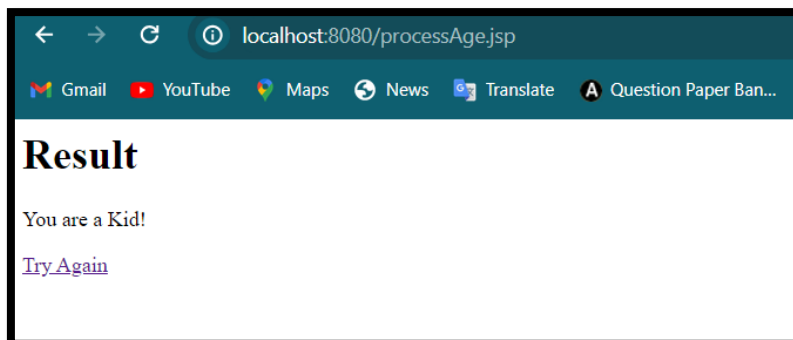


← → ↻ ⓘ localhost:8080/index.jsp

Gmail YouTube Maps News Translate Question Paper Ban...

Age Input

Enter your age:



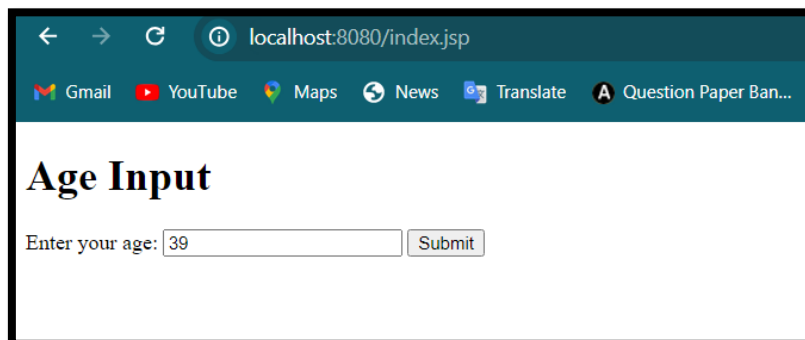
← → ↻ ⓘ localhost:8080/processAge.jsp

Gmail YouTube Maps News Translate Question Paper Ban...

Result

You are a Kid!

[Try Again](#)

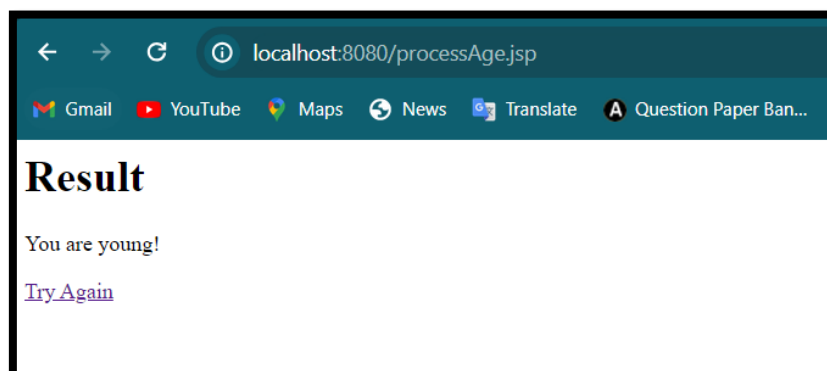


← → ↻ ⓘ localhost:8080/index.jsp

Gmail YouTube Maps News Translate Question Paper Ban...

Age Input

Enter your age:



← → ↻ ⓘ localhost:8080/processAge.jsp







Gmail YouTube Maps News Translate Question Paper Ban...

Result

You are young!

[Try Again](#)







← → ↻ ⓘ localhost:8080/index.jsp

 Gmail  YouTube  Maps  News  Translate  Question Paper Ban...

Age Input

Enter your age:

← → ↻ ⓘ localhost:8080/processAge.jsp

 Gmail  YouTube  Maps  News  Translate  Question Paper Ban...

Result

You are old!

[Try Again](#)

PROGRAM – 20

Objective:-

Write a program to display “Hello” using servlet.

Program:-

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

public class HelloWorld extends HttpServlet {

    private String message;

    public void init() throws ServletException {
        message = "Hello"; // Initialization
    }

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

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

        PrintWriter out = response.getWriter(); // Get writer

        out.println("<h1>" + message + "</h1>"); // Output message
    }

    public void destroy() {
        // Cleanup (optional)
    }
}
```

For deployment:

- Place the compiled `.class` file (e.g., `HelloWorld.class`) in `<Tomcat-installation-directory>/webapps/ROOT/WEB-INF/classes`.
- Configure the servlet in the `web.xml` file located at `<Tomcat-installation-directory>/webapps/ROOT/WEB-INF/` by defining a `<servlet>` and a `<servlet-mapping>` for the servlet class. In this case, the servlet name is "HelloWorld" and it's mapped to the URL pattern "/HelloWorld".

Xml

```
<servlet>
  <servlet-name>Hello</servlet-name>
  <servlet-class>HelloWorld</servlet-class>
</servlet>
```

```
<servlet-mapping>  
  <servlet-name>Hello</servlet-name>  
  <url-pattern>/Hello</url-pattern</servlet-mapping>
```

OUTPUT

