1. A web page is created using HTML, consists of various divisions which includes, head, body and footer parts. Write a program which includes various tags used inside the header tag.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<!-- Metadata -->

<meta charset="UTF-8">

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

<meta name="description" content="Sample web page demonstrating header tags in HTML">

<meta name="author" content="Your Name">

<!-- Title -->

<title>HTML Head Tag Example</title>

<!-- External CSS File -->

<link rel="stylesheet" href="style.css">

<!-- Internal CSS -->

<style>

body {

font-family: Arial, sans-serif;

background-color: #f4f4f4;

color: #333;

}

</style>

<!-- JavaScript -->

<script>

function showMessage() {

alert("Welcome to the Head Tag Example Page!");

}

</script>

</head>

<body onload="showMessage()">

<header>

<h1>Header Section</h1>

<p>This section is styled and controlled using tags from the &lt;head&gt; tag.</p>

</header>

<main>

<p>This is the body of the page.</p>

</main>

<footer>

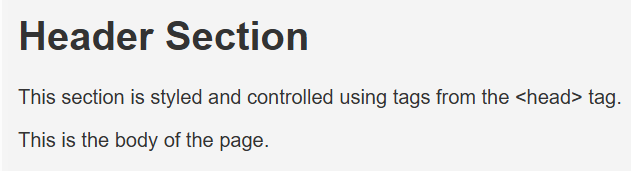
<p>© 2025 Example Page</p>

</footer>

</body>

</html>

Output:



1. Write a Javascript program to get the numbers from the user and display only the nearest odd numbers.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Nearest Odd Number</title>

<script>

function findNearestOdd() {

// Get input from user

let num = parseInt(document.getElementById("numberInput").value);

// Check if input is a number

if (isNaN(num)) {

alert("Please enter a valid number.");

return;

}

// Find nearest odd

let nearestOdd = (num % 2 === 0) ? num - 1 : num;

// Display result

document.getElementById("result").textContent = "Nearest odd number is: " + nearestOdd;

}

</script>

</head>

<body>

<h2>Find Nearest Odd Number</h2>

<label for="numberInput">Enter a number:</label>

<input type="number" id="numberInput">

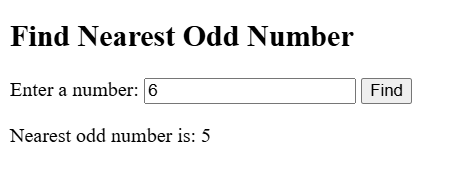
<button onclick="findNearestOdd()">Find</button>

<p id="result"></p>

</body>

</html>

Output:



1. Create a HTML web page, change the position of the element relative to the parent element and relative to itself using relative positioning with a CSS file.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Relative Positioning Example</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f4f4f4;

padding: 20px;

}

.parent {

position: relative;

width: 500px;

height: 300px;

background-color: #dfe9f3;

border: 2px solid #333;

padding: 20px;

margin: auto;

}

.child {

width: 250px;

padding: 10px;

margin: 10px 0;

background-color: #fff;

border: 1px solid #666;

position: relative;

}

/\* Moves the element relative to itself \*/

.self-relative {

top: 10px;

left: 20px;

}

/\* Moves the element further relative to the parent \*/

.relative-to-parent {

top: 40px;

left: 100px;

}

</style>

</head>

<body>

<div class="parent">

<h2>Parent Container</h2>

<div class="child self-relative">

This element is positioned <strong>relative to itself</strong>.

</div>

<div class="child relative-to-parent">

This element is <strong>offset more within the parent</strong>.

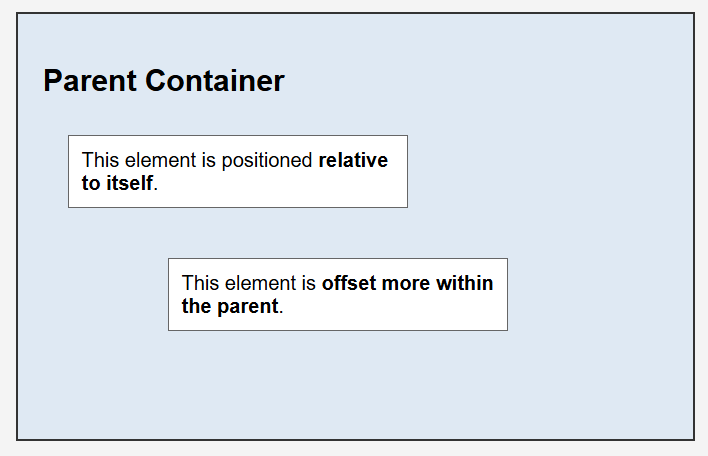
</div>

</div>

</body>

</html>

Output:



1. Write a Java servlet program to get the student registration details and display the same with the same servlet program.

Code:

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class StudentServlet extends HttpServlet {

// Handles both form display and submission

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.println("<html><head><title>Student Registration</title></head><body>");

out.println("<h2>Student Registration Form</h2>");

out.println("<form method='post' action=''>");

out.println("Name: <input type='text' name='name'><br><br>");

out.println("Email: <input type='email' name='email'><br><br>");

out.println("Course: <input type='text' name='course'><br><br>");

out.println("Age: <input type='number' name='age'><br><br>");

out.println("<input type='submit' value='Register'>");

out.println("</form></body></html>");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

String name = request.getParameter("name");

String email = request.getParameter("email");

String course = request.getParameter("course");

String age = request.getParameter("age");

out.println("<html><head><title>Registration Details</title></head><body>");

out.println("<h2>Student Registration Details</h2>");

out.println("<p><strong>Name:</strong> " + name + "</p>");

out.println("<p><strong>Email:</strong> " + email + "</p>");

out.println("<p><strong>Course:</strong> " + course + "</p>");

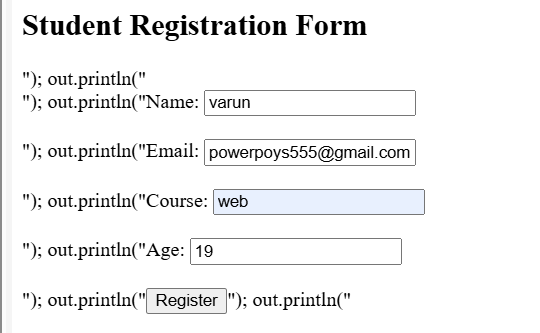
out.println("<p><strong>Age:</strong> " + age + "</p>");

out.println("</body></html>");

}

}

Output:



1. Make a registration form and help the organization to collect their employee’s details like, employer’s name, id, address, department, mobile number and select their city from the drop down list box using HTML and CSS.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Employee Registration Form</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #eef2f3;

padding: 20px;

}

.form-container {

background-color: #fff;

width: 400px;

margin: auto;

padding: 25px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0,0,0,0.1);

}

h2 {

text-align: center;

margin-bottom: 20px;

}

label {

display: block;

margin-top: 10px;

font-weight: bold;

}

input[type="text"],

input[type="number"],

textarea,

select {

width: 100%;

padding: 8px;

margin-top: 5px;

border: 1px solid #ccc;

border-radius: 4px;

}

input[type="submit"] {

background-color: #007bff;

color: white;

padding: 10px;

width: 100%;

border: none;

border-radius: 4px;

margin-top: 15px;

cursor: pointer;

}

input[type="submit"]:hover {

background-color: #0056b3;

}

</style>

</head>

<body>

<div class="form-container">

<h2>Employee Registration</h2>

<form>

<label for="name">Employee Name:</label>

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

<label for="empid">Employee ID:</label>

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

<label for="address">Address:</label>

<textarea id="address" name="address" rows="3" required></textarea>

<label for="department">Department:</label>

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

<label for="mobile">Mobile Number:</label>

<input type="number" id="mobile" name="mobile" required>

<label for="city">City:</label>

<select id="city" name="city" required>

<option value="">--Select City--</option>

<option value="Delhi">Delhi</option>

<option value="Mumbai">Mumbai</option>

<option value="Bangalore">Bangalore</option>

<option value="Hyderabad">Hyderabad</option>

<option value="Chennai">Chennai</option>

</select>

<input type="submit" value="Register">

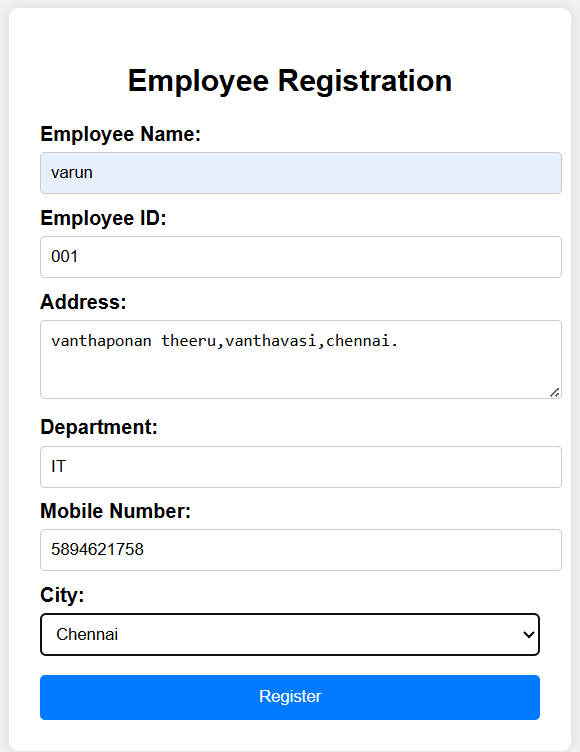
</form>

</div>

</body>

</html>

Output:



1. Write a HTML program for numerous elements and that should be organized separately with class selector by using Cascading Style Sheets(CSS) for external style sheets.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>HTML Elements with CSS Classes</title>

<link rel="stylesheet" href="styles.css"> <!-- External CSS -->

</head>

<body>

<div class="header">

<h1>Welcome to Our Sample Page</h1>

<p>This page demonstrates multiple HTML elements styled using class selectors.</p>

</div>

<div class="section paragraph-section">

<h2>Paragraphs</h2>

<p class="text-block">This is the first paragraph. It contains some basic content to show formatting.</p>

<p class="text-block">Here's another paragraph with more text to demonstrate spacing and font styling.</p>

</div>

<div class="section image-section">

<h2>Image Section</h2>

<img class="image" src="https://via.placeholder.com/300" alt="Sample Image">

</div>

<div class="section list-section">

<h2>Lists</h2>

<ul class="list">

<li>First item</li>

<li>Second item</li>

<li>Third item</li>

</ul>

</div>

<div class="section table-section">

<h2>Table</h2>

<table class="table">

<tr>

<th>Name</th>

<th>Age</th>

<th>City</th>

</tr>

<tr>

<td>Alice</td>

<td>24</td>

<td>New York</td>

</tr>

<tr>

<td>Bob</td>

<td>30</td>

<td>Los Angeles</td>

</tr>

</table>

</div>

<div class="section form-section">

<h2>Contact Form</h2>

<form class="form">

<label for="name">Name:</label>

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

<label for="email">Email:</label>

<input type="email" id="email" name="email"><br>

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

</form>

</div>

<div class="footer">

<p>© 2025 Sample Page. All rights reserved.</p>

</div>

</body>

</html>

Output:



1. Create a HTML web page, change the position of the element float to get two elements to float over each other using float positioning with a CSS file.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Float Positioning Example</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f4f4f4;

padding: 20px;

}

.container {

width: 100%;

overflow: hidden; /\* Clearfix \*/

}

.box1, .box2 {

width: 200px;

height: 200px;

padding: 20px;

color: white;

text-align: center;

font-size: 20px;

font-weight: bold;

border-radius: 10px;

}

.box1 {

background-color: #3498db;

float: left;

margin-right: 20px; /\* Space between boxes \*/

}

.box2 {

background-color: #e74c3c;

float: left;

}

/\* Clear floats after the boxes \*/

.container::after {

content: "";

display: table;

clear: both;

}

</style>

</head>

<body>

<h2>Float Positioning Example</h2>

<div class="container">

<div class="box1">Box 1 (Float Left)</div>

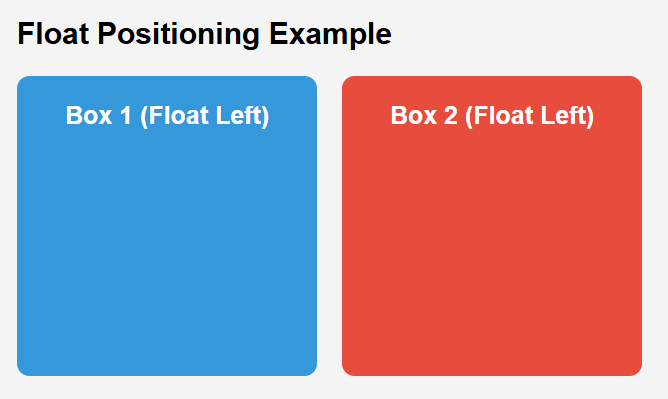
<div class="box2">Box 2 (Float Left)</div>

</div>

</body>

</html>

Output:



1. Write a java script page to find the • Exponential value. • Limit the number of digits after the decimal value. • Convert a number to string. • Return negative infinity using the number object in the HTML document.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>JavaScript Number Operations</title>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

}

h2 {

color: #3498db;

}

.output {

background-color: #f0f8ff;

padding: 10px;

margin: 5px 0;

border: 1px solid #ddd;

}

</style>

</head>

<body>

<h2>JavaScript Number Operations</h2>

<p id="exponential"></p>

<p id="limitedDecimal"></p>

<p id="numberToString"></p>

<p id="negativeInfinity"></p>

<script>

// 1. Find the Exponential Value of a number

const base = 5;

const exponent = 3;

const exponentialValue = Math.pow(base, exponent);

document.getElementById("exponential").innerHTML = `Exponential value of ${base} raised to the power of ${exponent} is: ${exponentialValue}`;

// 2. Limit the number of digits after the decimal value

const numberWithDecimals = 123.456789;

const limitedDecimal = numberWithDecimals.toFixed(2); // Limiting to 2 decimal places

document.getElementById("limitedDecimal").innerHTML = `Limited decimal value (2 digits) of ${numberWithDecimals} is: ${limitedDecimal}`;

// 3. Convert a number to string

const number = 98765;

const numberAsString = number.toString();

document.getElementById("numberToString").innerHTML = `Number ${number} as a string is: "${numberAsString}"`;

// 4. Return Negative Infinity using the Number object

const negativeInfinity = Number.NEGATIVE\_INFINITY;

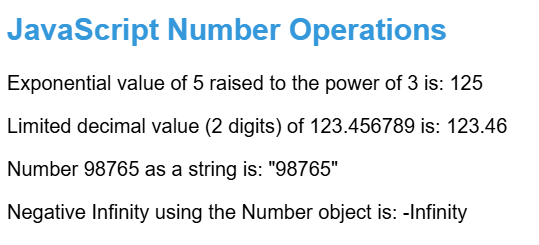
document.getElementById("negativeInfinity").innerHTML = `Negative Infinity using the Number object is: ${negativeInfinity}`;

</script>

</body>

</html>

Output:



1. Write a JavaScript program for Email validation. You can apply validation on the below text questions to ensure they accept answers in a specific format.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Email Validation</title>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

}

h2 {

color: #3498db;

}

.output {

background-color: #f0f8ff;

padding: 10px;

margin: 5px 0;

border: 1px solid #ddd;

}

.error {

color: red;

}

</style>

</head>

<body>

<h2>Email Validation Example</h2>

<form id="emailForm" onsubmit="return validateEmail()">

<label for="email">Enter your email:</label><br><br>

<input type="text" id="email" name="email" required><br><br>

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

</form>

<p id="message"></p>

<script>

// Email Validation function

function validateEmail() {

// Get the email input value

const email = document.getElementById("email").value;

// Regular expression for validating the email format

const emailPattern = /^[a-zA-Z0-9.\_-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}$/;

// Get the message element to display the result

const messageElement = document.getElementById("message");

// Validate the email with the regex

if (emailPattern.test(email)) {

messageElement.textContent = "Email is valid!";

messageElement.classList.remove("error");

messageElement.classList.add("output");

} else {

messageElement.textContent = "Invalid email address. Please enter a valid email.";

messageElement.classList.remove("output");

messageElement.classList.add("error");

return false; // Prevent form submission

}

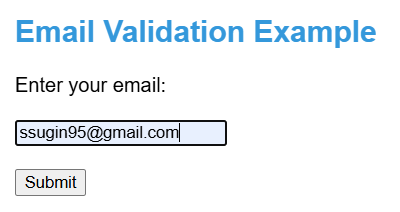
}

</script>

</body>

</html>

Output:



1. Write a HTML program to add a background GIF to a webpage and alter its height, width, transparency and transition using Cascading Style Sheets.

Code:

**background\_gif.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Background GIF with CSS</title>

<style>

body {

margin: 0;

padding: 0;

height: 100vh;

}

.background-container {

background-image: url('your-gif.gif'); /\* Use the path to your GIF file \*/

background-size: cover; /\* Make the background cover the entire screen \*/

background-position: center;

height: 100vh;

width: 100%;

transition: opacity 2s ease-in-out; /\* Smooth transition for opacity \*/

opacity: 0.7; /\* Initial transparency \*/

}

/\* On hover, change the transparency \*/

.background-container:hover {

opacity: 1; /\* Full opacity on hover \*/

}

</style>

</head>

<body>

<div class="background-container">

<!-- You can add content here if you wish -->

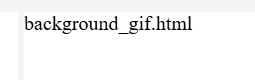
<h1 style="color: white; text-align: center; padding-top: 40%;">Welcome to the Animated Background!</h1>

</div>

</body>

</html>

Output:



c