1. **Write a program which consists of single line and multiline comments.**

**Single line**

<!DOCTYPE html>

<html>

<body>

<h1 id="myH"></h1>

<p id="myP"></p>

<script>

// Change heading:

document.getElementById("myH").innerHTML = "JavaScript Comments";

// Change paragraph:

document.getElementById("myP").innerHTML = "My first paragraph.";

</script>

</body>

</html>

**Output**

**JavaScript Comments**

My first paragraph.

**Multiline comments**

<!DOCTYPE html>

<html>

<body>

<h1 id="myH"></h1>

<p id="myP"></p>

<script>

/\*

The code below will change

the heading with id = "myH"

and the paragraph with id = "myP"

\*/

document.getElementById("myH").innerHTML = "JavaScript Comments";

document.getElementById("myP").innerHTML = "My first paragraph.";

</script>

</body>

</html>

**Output**

**JavaScript Comments**

My first paragraph.

**Create an array consisting of fruits names and when we click on a button sort it in descending order.**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>JavaScript Sort an Array Alphabetically</title>

</head>

<body>

<script>

var fruits = ["Banana", "Orange", "Apple", "Papaya", "Mango"];

var sorted = fruits.sort();

document.write(fruits + "<br>"); // Outputs: Apple,Banana,Mango,Orange,Papaya

document.write(sorted); // Outputs: Apple,Banana,Mango,Orange,Papaya

</script>

</body>

</html>

**Output**

Apple,Banana,Mango,Orange,Papaya  
Apple,Banana,Mango,Orange,Papaya

**Write a program with FOR/IN loop.**

<!DOCTYPE html>

<html>

<body>

<h2>JavaScript For Loop</h2>

<p id="demo"></p>

<script>

var cars = ["BMW", "Volvo", "Saab", "Ford", "Fiat", "Audi"];

var text = "";

var i;

for (i = 0; i < cars.length; i++) {

text += cars[i] + "<br>";

}

document.getElementById("demo").innerHTML = text;

</script>

</body>

</html>

**Output**

**JavaScript For Loop**

BMW  
Volvo  
Saab  
Ford  
Fiat  
Audi

**Create an object "person" with properties firstname and lastname. Display these properties using 2 different ways of accessing.**

<!DOCTYPE html>

<html>

<body>

<h1>Demo: JavaScript Object</h1>

<p id="p1"></p>

<p id="p2"></p>

<p id="p3"></p>

<p id="p4"></p>

<p id="p5"></p>

<script>

var person = {

firstName: "James",

lastName: "Bond",

age: 25,

getFullName: function () {

return this.firstName + ' ' + this.lastName

}

};

document.getElementById("p1").innerHTML = person.firstName;

document.getElementById("p2").innerHTML = person.lastName;

document.getElementById("p3").innerHTML = person["firstName"];

document.getElementById("p4").innerHTML = person["lastName"];

document.getElementById("p5").innerHTML = person.getFullName();

</script>

</body>

</html>

**Output**

**Demo: JavaScript Object**

James

Bond

James

Bond

James Bond

**Write a program with variable hoisting (initialization first then declaration).**

<!DOCTYPE html>

<html>

<body>

<p id="demo"></p>

<script>

var x = 5; // Initialize x

var y = 7; // Initialize y

elem = document.getElementById("demo"); // Find an element

elem.innerHTML = x + " " + y; // Display x and y

</script>

</body>

</html>

**Output**

5 7

**Use strict mode in your program and understand why variable not declared causes error.**

<!DOCTYPE html>

<html>

<body>

<h2>Global "use strict" declaration.</h2>

<p>Activate debugging in your browser (F12) to see the error report.</p>

<script>

"use strict";

myFunction();

function myFunction() {

y = 3.14; // This will cause an error (y is not defined)

}

</script>

</body>

</html>

**Output**

**Global "use strict" declaration.**

Activate debugging in your browser (F12) to see the error report.

**Write program using this keyword when we click on button, it should be disappear.**

<!DOCTYPE html>

<html>

<body>

<p id="demo">Click me to change my HTML content (innerHTML).</p>

<p> this should disappear</p>

<button onclick="window.open();">open button</button>

<button onclick="window.close();">close button</button>

</body>

</html>

**Output**

Click me to change my HTML content (innerHTML).

this should disappear

open button close button

**Invoke a function using call() and apply(). Also Spot the difference between them.**

**call()**

<!DOCTYPE html>

<html>

<head>

<title>call() method</title>

</head>

<body style = "text-align:center;">

<h1 style = "color:green;" >

FrontEnd

</h1>

<button onClick="fun()">

click

</button>

<p id="FE"></p>

<!-- Script to use call() method to call

function -->

<script>

function fun() {

let p = {

fullName: function(addr1, addr2) {

return this.fName + " " + this.lName

+ ", " + addr1 + ", " + addr2;

}

}

let p1 = {

fName:"FEfName",

lName: "FElName",

}

let x = p.fullName.call(p1, "India", "USA");

document.getElementById("FE").innerHTML = x;

}

</script>

</body>

</html>

**Output**

**Before clicking the button:**

**FrontEnd**

click

**After clicking the button:**

**FrontEnd**

click

FEfName FElName, India, USA

**apply()**

<!DOCTYPE html>

<html>

<head>

<title>JavaScript apply() method</title>

</head>

<body style = "text-align:center;">

<h1 style = "color:green;" >

BackEnd

</h1>

<button onClick="fun()">

click

</button>

<p id="GFG"></p>

<script>

function fun() {

let p = {

fullName: function(addr1, addr2) {

return this.fName + " " + this.lName

+ ", " + addr1 + ", " + addr2;

}

}

let p1 = {

fName:"BEName",

lName: "BEName",

}

let x = p.fullName.apply(p1, ["Japan", "Europe"]);

document.getElementById("GFG").innerHTML = x;

}

</script>

</body>

</html>

**Output**

**Before clicking the button:**

**BackEnd**

click

**After clicking the button:**

**BackEnd**

click

BEName BEName, Japan, Europe

**Write a program for COUNTER. Whenever we click on a "Count" button, value should increment.**

<script type="text/javascript">

// if the count variable is undefined, set its value to zero

if(! count) {

var count = 0;

}

// function to increment value of count variable

incrementCount() {

count++;

}

</script>

<!-- button to call incrementCount() function, on click -->

<form>

<input type="button" value="Count" onclick="incrementCount()">

</form>

<br />

<script type="text/javascript">

// output count variable

document.write(count);

</script>

**Output**

Top of Form

Bottom of Form

count

**Create a Counter with the help of getter and setter accessors.**

<!DOCTYPE html>

<html>

<body>

<h2>JavaScript Getters and Setters</h2>

<p>Getters and setters allow you to get and set object properties via methods.</p>

<p>This example uses a lang property to get the value of the language property:</p>

<p id="demo"></p>

<script>

// Create an object:

const person = {

firstName: "John",

lastName: "Doe",

language: "en",

get lang() {

return this.language;

}

};

// Display data from the object using a getter:

document.getElementById("demo").innerHTML = person.lang;

</script>

</body>

</html>

**Output**

**JavaScript Getters and Setters**

Getters and setters allow you to get and set object properties via methods.

This example uses a lang property to get the value of the language property:

en

**Add some more properties to an existing function using object prototypes.**

<!DOCTYPE html>

<html>

<body>

<h2>JavaScript Objects</h2>

<p>Using an object constructor:</p>

<p id="demo"></p>

<script>

function Person(first, last, age, eye) {

this.firstName = first;

this.lastName = last;

this.age = age;

this.eyeColor = eye;

}

const myFather = new Person("John", "Doe", 50, "blue");

const myMother = new Person("Sally", "Rally", 48, "green");

document.getElementById("demo").innerHTML =

"My father is " + myFather.age + ". My mother is " + myMother.age;

</script>

</body>

</html>

**Output**

**JavaScript Objects**

Using an object constructor:

My father is 50. My mother is 48

**Add an event listener to a button. When we click on it should display current date and time.**

<!DOCTYPE html>

<html>

<body>

<p>Click the button to display the time.</p>

<button onclick="getElementById('demo').innerHTML=Date()">What is the time?</button>

<p id="demo"></p>

</body>

</html>

**Output**

**Before clicking the button:**

Click the button to display the time.

What is the time?

**After clicking the button:**

Click the button to display the time.

What is the time?

Sun Jun 13 2021 18:33:50 GMT+0530 (India Standard Time)

**Show the javascript validation. When you click submit, the text box doesn't need to be empty.**

function required()

{

var empt = document.forms["form1"]["text1"].value;

if (empt == "")

{

alert("Please input a Value");

return false;

}

else

{

alert('Code has accepted : you can try another');

return true;

}

}

**How console.log() can be used for Debugging.**

<!DOCTYPE html>

<html>

<body>

<h2>Web Page</h2>

<p>Activate debugging in your browser (Chrome, IE, Firefox) with F12, and select "Console" in the debugger menu.</p>

<script>

a = 5;

b = 6;

c = a + b;

console.log(c);

</script>

</body>

</html>

**Output**

**Web Page**

Activate debugging in your browser (Chrome, IE, Firefox) with F12, and select "Console" in the debugger menu.

**Write functions to set a cookie, get a cookie and check a cookie in a single program.**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>Set and Get Cookies in JavaScript</title>

</head>

<body>

<script>

// A custom function to set cookies

function setCookie(name, value, daysToLive) {

// Encode value in order to escape semicolons, commas, and whitespace

var cookie = name + "=" + encodeURIComponent(value);

if(typeof daysToLive === "number") {

/\* Sets the max-age attribute so that the cookie expires

after the specified number of days \*/

cookie += "; max-age=" + (daysToLive\*24\*60\*60);

document.cookie = cookie;

}

}

// A custom function to get cookies

function getCookie(name) {

// Split cookie string and get all individual name=value pairs in an array

var cookieArr = document.cookie.split(";");

// Loop through the array elements

for(var i = 0; i < cookieArr.length; i++) {

var cookiePair = cookieArr[i].split("=");

/\* Removing whitespace at the beginning of the cookie name

and compare it with the given string \*/

if(name == cookiePair[0].trim()) {

// Decode the cookie value and return

return decodeURIComponent(cookiePair[1]);

}

}

// Return null if not found

return null;

}

// A custom function to check cookies

function checkCookie() {

// Get cookie using our custom function

var firstName = getCookie("firstName");

if(firstName != null) {

alert("Welcome again, " + firstName);

} else {

firstName = prompt("Please enter your first name:");

if(firstName != "" && firstName != null) {

// Set cookie using our custom function

setCookie("firstName", firstName, 1);

}

}

}

// Check the cookie on page load

window.onload = checkCookie;

// Uncomment the following line to delete this cookie

// setCookie("firstName", "", 0);

</script>

<p><strong>Note:</strong> Now if you press the "Show Output" button or refresh the page, you will see a greeting message, if you've entered your first name in the prompt dialog box. See the last line of the JavaScript code to know how to delete this <b>firstName</b> cookie.</p>

</body>

</html>

**Output**

**Note:** Now if you press the "Show Output" button or refresh the page, you will see a greeting message, if you've entered your first name in the prompt dialog box. See the last line of the JavaScript code to know how to delete this **firstName** cookie.

**Create a JSON object and access it using dot notation.**

<!DOCTYPE html>

<html>

<body>

<h2>Creating an Object from a JSON Literal</h2>

<p id="demo"></p>

<script>

const myObj = {"name":"John", "age":30, "car":null};

document.getElementById("demo").innerHTML = myObj.name;

</script>

</body>

</html>

**Output**

**Creating an Object from a JSON Literal**

John