Module 6) JAVASCRIPT BASIC & DOM

**Que - 1 : What is JavaScript?**

**Ans.**

        JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications. It is complementary to and integrated with Java. JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform. Javascript is the most popular programming language in the world and that makes it a programmer’s great choice.

**Que -  2 : What is the use of isNaN function?**

**Ans.**   .

          The isNaN() function determines whether a value is an illegal number (Not-a-Number).

       This function returns true if the value equates to NaN. Otherwise it returns false.

       This function is different from the Number specific Number.isNaN() method.

       The global isNaN() function, converts the tested value to a Number, then tests it.

        Number.isNaN() does not convert the values to a Number, and will not return true for any value that is not of the type Number.

**Que - 3   What is negative Infinity?**

**Ans.**

               The negative infinity in JavaScript is a constant value that is used to represent a value that is the lowest available. This means that no other number is less than this value. It can be generated using a self-made function or by an arithmetic operation.

Note: JavaScript shows the NEGATIVE\_INFINITY value as -Infinity.

Negative infinity is different from mathematical infinity in the following ways:

* Negative infinity results in -0(different from 0 ) when divided by any other number.
* When divided by itself or positive infinity, negative infinity return NaN
* Negative infinity, when divided by any positive number (apart from positive infinity) is negative infinity.
* Negative infinity, divided by any negative number (apart from negative infinity) is positive infinity.
* If we multiply negative infinity with NaN, we will get NaN as a result.
* The product of 0 and negative infinity is Nan.
* The product of two negative infinities is always a positive infinity.
* The product of both positive and negative infinity is always negative infinity.

**Syntax :**

<!DOCTYPE html>

<html>

<body>

<h1>JavaScript Numbers</h1>

<h2>The NEGATIVE\_INFINITY Property</h2>

<p>Create a negative infinity:</p>

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

<script>

let n = (-Number.MAX\_VALUE) \* 2;

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

</script>

</body>

</html>

**Que - 4   Which company developed JavaScript?**

**Ans.**

            JavaScript was invented by Brendan Eich in 1995. It was developed for Netscape 2, and became the ECMA-262 standard in 1997. After Netscape handed JavaScript over to ECMA, the Mozilla foundation continued to develop JavaScript for the Firefox browser.

**Que - 5  What are undeclared and undefined variables?**

**Ans.**

**Undeclared variables :**

          A variable is undeclared if it has not been declared with an appropriate keyword (i.e. var, let or const). Accessing an undeclared variable will throw a [ReferenceError](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/ReferenceError).

**Example:**

         console.log(x); // ReferenceError: x is not defined

**Undefined variables :**

          A variable is undefined if it hasn't been assigned a value. undefined is a primitive data type in JavaScript and represents the absence of a value, intentional or otherwise.

**Example:**

           let x;

        console.log(x); // undefined

**Que - 6  Write the code for adding new elements dynamically?**

**Ans.**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

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

**<title>Document</title>**

**</head>**

**<body>**

**<h3>How to dynamically create new elements in JavaScript?</h3>**

**<div id="container">**

**<!-- Newly created elements will be appended here -->**

**</div>**

**<button onclick="createNewElement()">Create Element</button>**

**<script>**

**function createNewElement() {**

**// Create a new paragraph element**

**var newParagraph = document.createElement('h1');**

**// Set the text content of the paragraph**

**newParagraph.innerHTML = 'This is a dynamically created paragraph.';**

**// Append the paragraph to the container div**

**var container = document.getElementById('container');**

**container.prepend(newParagraph);**

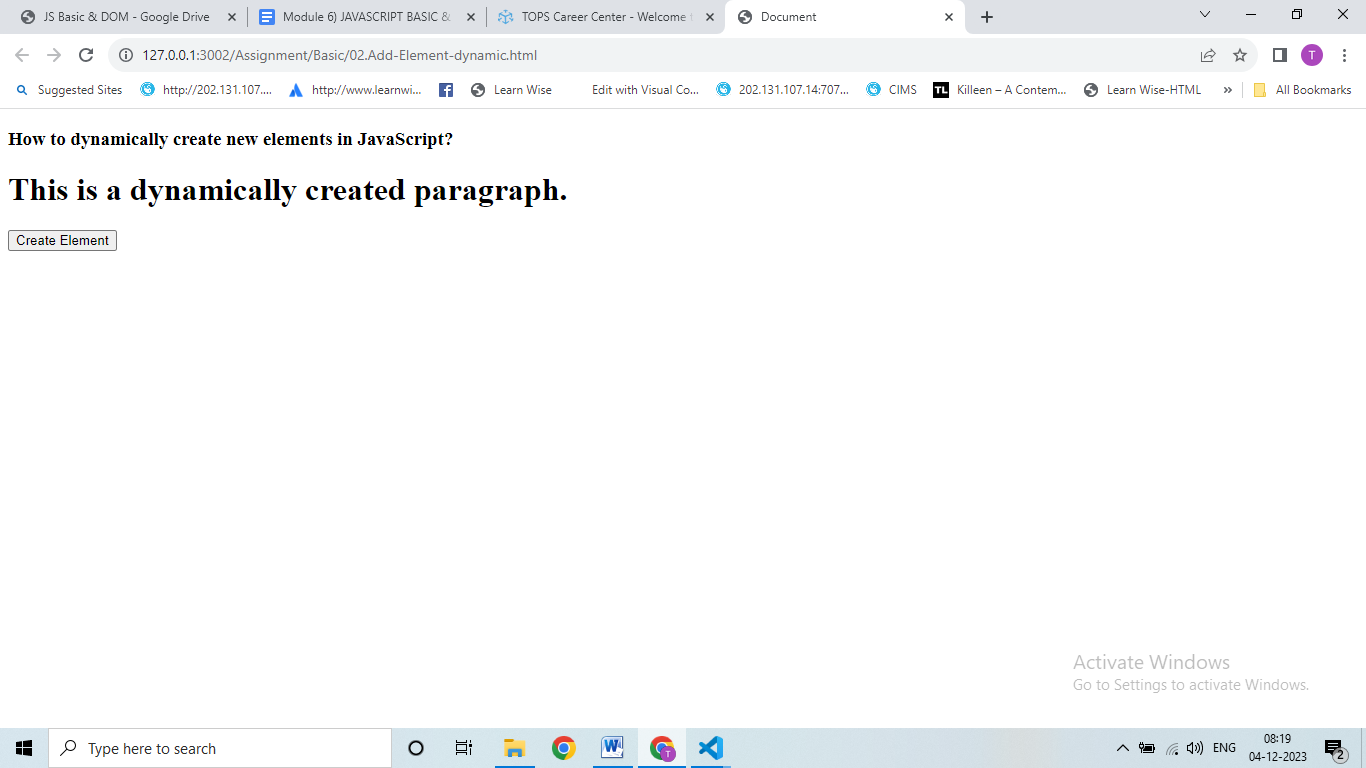
**}**

**</script>**

**</body>**

**</html**

**OUTPUT:**



**Que - 7  What is the difference between ViewState and SessionState?**

**Ans.**

**Usage:**

* SessionState: It can be used to store information that you wish to access on different web pages.

* ViewState : It can be used to store information that you wish to access from the same web page.

**Differences between ViewState and SessionState:**

|  |  |
| --- | --- |
| **ViewState** | **SessionState** |
| Maintained at page level only. | Maintained at session level. |
| View state can only be visible from a single page and not multiple pages. | Session state value availability is across all pages available in a user session. |
| It will retain values in the event of a postback operation occurring. | In session state, user data remains in the server. Data is available to the user until the browser is closed or there is session expiration. |
| Information is stored on the client’s end only. | Information is stored on the server. |
| used to allow the persistence of page-instance-specific data. | used for the persistence of user-specific data on the server’s end. |
| ViewState values are lost/cleared when a new page is loaded. | SessionState can be cleared by programmer or user or in case of timeouts. |
|  |  |

**Que - 8  What is === operator?**

**Ans.**

=== (Triple equals) is a strict equality comparison operator in JavaScript, which returns false for the values which are not of a similar type. This operator performs type casting for equality. If we compare 2 with “2” using ===, then it will return a false value.

**How === Works Exactly?**

* Strict equality === checks that two values are the same or not.
* Values are not implicitly converted to some other value before comparison.
* If the variable values are of different types, then the values are considered as unequal.
* If the variables are of the same type, are not numeric, and have the same value, they are considered as equal.
* Lastly, If both variable values are numbers, they are considered equal if both are not NaN (Not a Number) and are the same value.

**Que - 9  How can the style/class of an element be changed?**

**Ans.**

       To change the style or class of an element in JavaScript, you can use the following methods:

**Changing style:**

var element = document.getElementById("myElement");

element.style.property = "value";

**Changing class:**

var element = document.getElementById("myElement");

element.className = "newClassName";

**Que - 10  How to read and write a file using JavaScript?**

**Ans.**

       Reading and writing files using JavaScript is typically not possible in a web browser environment for security reasons. However, in server-side JavaScript (e.g., Node.js), you can use modules like fs for file I/O operations.

**Que - 11  What are all the looping structures in JavaScript?**

**Ans.**

         Looping in programming languages is a feature that facilitates the execution of a set of instructions/functions repeatedly while some condition evaluates to true. For example, suppose we want to print “Hello World” 10 times.

**There are mainly two types of loops:**

* **Entry Controlled Loop:** In these types of loops, the test condition is tested before entering the loop body. The **for Loop** and **while Loop** are entry-controlled loops.

* **Exit Controlled Loop:** In these types of loops the test condition is tested or evaluated at the end of the loop body. Therefore, the loop body will execute at least once, irrespective of whether the test condition is true or false. The **do-while loop** is an exit -controlled loop.

**Que - 12  How can you convert the string of any base to an integer in JavaScript?**

**Ans.**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Document</title>

</head>

<body>

    <script>

        // with using number method

        var stringtonumber = "100";

        document.write(Number(stringtonumber));

        // with using parseInt

        var stringtonumber1 = "152";

        document.write(parseInt(stringtonumber1));

    </script>

</body>

</html>

**Que - 13  What is the function of the delete operator?**

**Ans.**

      The JavaScript [pop()](https://www.geeksforgeeks.org/javascript-array-pop-method/), [shift()](https://www.geeksforgeeks.org/javascript-array-shift-method/), or [splice()](https://www.geeksforgeeks.org/javascript-array-splice-method/) methods are available to delete an element from an array. But because of the key-value pair in an object, deleting is more complicated. Note that the delete operator only works on objects and not on variables or functions.

**Que - 14  What are all the types of Pop up boxes available in JavaScript?**

**Ans.**

      In Javascript, popup boxes are used to display the message or notification to the user.

There are three types of  Pop up boxes namely 1. Alert Box, 2. Confirm Box and  3.Prompt Box.

**Que - 15   What is the use of Void (0)?**

**Ans.**

               The javascript:void(0) is often used while creating HTML documents. This is used when an expression in the web page can lead to an undesired effect. This can be loading a new web page. In such situations, adding javascript:void(0) to the HTML document will remove this effect.

              The void operator returns a null value so that the browser does not load a new web page. This utility comes in handy when you want to perform useful actions on the web page. This can be updating certain values on the page by clicking on a link.

Example:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Document</title>

</head>

<body>

       <p>

        <p>Click the following , This won't react at all...</p>

       </p>

       <a href="javascript:void(alert('warning!!!'))">Click me!</a>

</body>

</html>

**Que - 16  How can a page be forced to load another page in JavaScript?**

**Ans.**

**You can force a page to load another page in JavaScript by setting the window.location property to the new URL:**

window.location.href = "newPage.html";

**Que - 17  What are the disadvantages of using innerHTML in JavaScript?**

**Ans.**

**Disadvantages of using innerHTML property in JavaScript:**

* The use of innerHTML is very slow: The process of using innerHTML is much slower as its contents are slowly built, also already parsed contents and elements are also re-parsed which takes time.

* Preserves event handlers attached to any DOM elements: The event handlers do not get attached to the new elements created by setting innerHTML automatically. To do so one has to keep track of the event handlers and attach it to new elements manually. This may cause a memory leak on some browsers.

* Content is replaced everywhere: Either you add, append, delete or modify contents on a webpage using innerHTML, all contents is replaced, also all the DOM nodes inside that element are reparsed and recreated.

* Old content replaced issue: The old content is replaced even if object.innerHTML = object.innerHTML + ‘html’ is used instead of object.innerHTML += ‘html’. There is no way of appending without reparsing the whole innerHTML. Therefore, working with innerHTML becomes very slow. String concatenation just does not scale when dynamic DOM elements need to be created as the plus’ and quote openings and closings become difficult to track.

* Can break the document: There is no proper validation provided by innerHTML, so any valid HTML code can be used. This may break the document of JavaScript. Even broken HTML can be used, which may lead to unexpected problems.

**Que - 18   Create password field with show hide functionalities**

**Ans.**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Document</title>

</head>

<body>

    <label for="">Enter Password : </label><input type="password" id="pwd" value="Hello1234">

    <br>

    <br>

    <input type="checkbox" onclick="ShowPassword()" name="" id="check"> <label for="">Password</label>

    <script>

        function ShowPassword() {

            var pwd = document.getElementById("pwd");

            if (pwd.type === "password") {

                pwd.type = "text";

            } else {

                pwd.type = "password"

            }

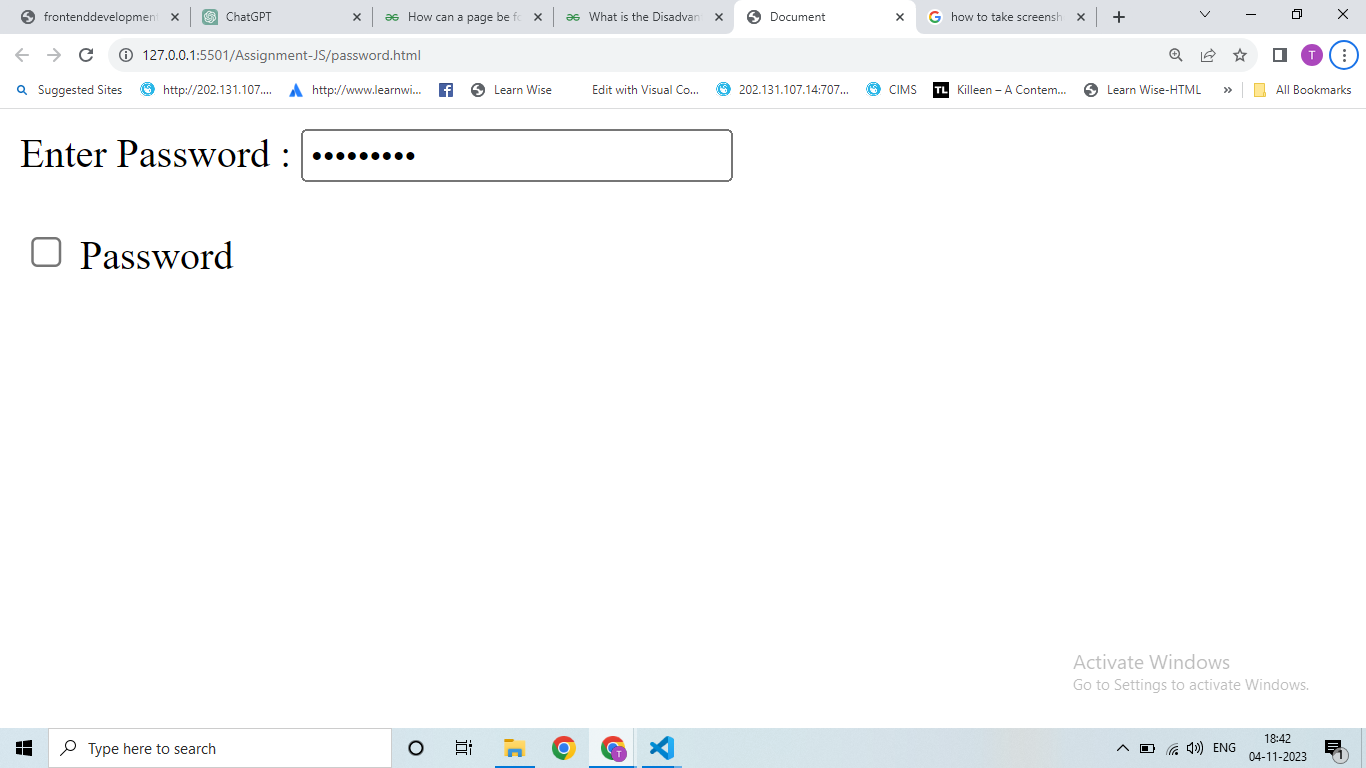
        }

    </script>

</body>

</html>

**OUTPUT:**

****

**Que - 19  Create basic math operation in JS**

**Ans.**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

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

**<title>Document</title>**

**</head>**

**<style>**

**#parent {**

**display: flex;**

**}**

**</style>**

**<body>**

**<h1>Maths Operations</h1>**

**<div>**

**<label for="">Enter 1st Number:</label>**

**<input type="number" id="value1">**

**</div>**

**<div>**

**<label for="">Enter 2nd Number:</label>**

**<input type="number" name="" id="value2">**

**</div>**

**<div id="parent">**

**<div>**

**<button onclick="caculator('+')">+</button>**

**<button onclick="caculator('-')">-</button>**

**<button onclick="caculator('\*')">\*</button> <br>**

**<button onclick="caculator('/')">/</button>**

**<button onclick="caculator('%')">%</button>**

**</div>**

**<div>**

**<p>Answer is : <span id="result">-</span></p>**

**</div>**

**</div>**

**<script>**

**function caculator(oparators) {**

**var value1 = parseFloat(document.getElementById("value1").value);**

**var value2 = parseFloat(document.getElementById("value2").value);**

**var result = document.getElementById("result");**

**if (isNaN(value1) || isNaN(value2)) {**

**result.textContent = "Invalid Data";**

**} else {**

**switch (oparators) {**

**case "+":**

**result.textContent = (value1 + value2);**

**break;**

**case "-":**

**result.innerHTML = (value1 - value2);**

**break;**

**case "\*":**

**result.innerHTML = (value1 \* value2);**

**break;**

**case "/":**

**result.innerHTML = (value1 / value2);**

**break;**

**case "%":**

**result.innerHTMLt = ((value1 / 100) \* value2);**

**break;**

**default:**

**result.innerHTML = "Invalid Operations"**

**break;**

**}**

**}**

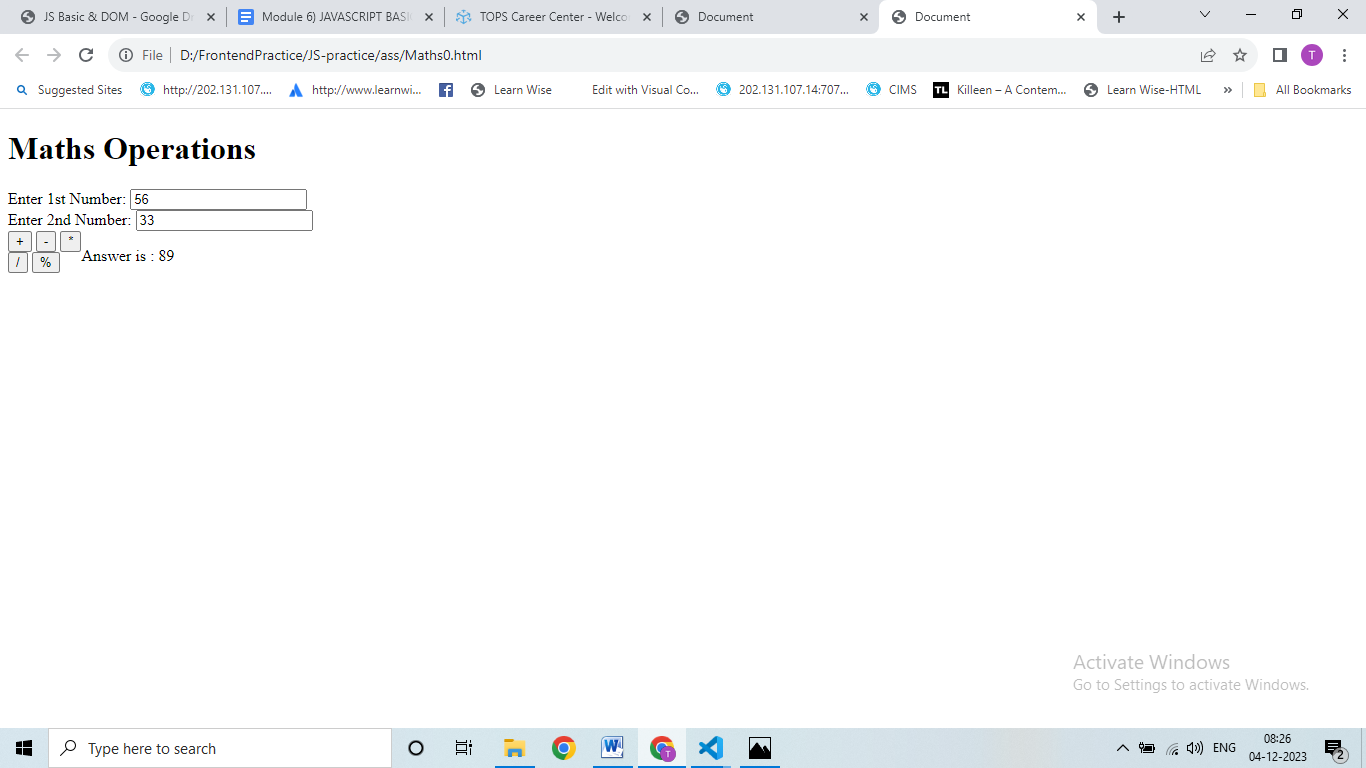
**};**

**</script>**

**</body>**

**</html>**

**OUTPUT:**



**Que - 20  Create result**

**Ans:**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

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

**<title>Document</title>**

**</head>**

**<body>**

**<fieldset>**

**<table align="center">**

**<h2 align="center">Marksheet for Information Technology</h2>**

**<p align="center">Enter Marks</p>**

**<tr>**

**<td>1.C Language</td>**

**<td><input type="number" id="subject1"></td>**

**</tr>**

**<tr>**

**<td>2.C++ Language</td>**

**<td><input type="number" id="subject2"></td>**

**</tr>**

**<tr>**

**<td>3.Database</td>**

**<td><input type="number" id="subject3"></td>**

**</tr>**

**<tr>**

**<td>4.HTML</td>**

**<td><input type="number" id="subject4"></td>**

**</tr>**

**<tr>**

**<td>5.CSS</td>**

**<td><input type="number" id="subject5"></td>**

**</tr>**

**<tr>**

**<td>6.php</td>**

**<td><input type="number" id="subject6"></td>**

**</tr>**

**<tr>**

**<td>7.Core java</td>**

**<td><input type="number" id="subject7"></td>**

**</tr>**

**<tr>**

**<td></td>**

**<td><button onclick="result()">Result</button></td>**

**</tr>**

**<tr>**

**<td>Total is: <span id="total"></span></td>**

**<td>Percentage is : <span id="percentage"></span></td>**

**</tr>**

**</table>**

**</fieldset>**

**<script>**

**function result() {**

**//get Subject Marks**

**var subject1 = parseFloat(document.getElementById("subject1").value) || 0;**

**var subject2 = parseFloat(document.getElementById("subject2").value) || 0;**

**var subject3 = parseFloat(document.getElementById("subject3").value) || 0;**

**var subject4 = parseFloat(document.getElementById("subject4").value) || 0;**

**var subject5 = parseFloat(document.getElementById("subject5").value) || 0;**

**var subject6 = parseFloat(document.getElementById("subject6").value) || 0;**

**var subject7 = parseFloat(document.getElementById("subject7").value) || 0;**

**//get total & percentage**

**var total = subject1 + subject2 + subject3 + subject4 + subject5 + subject6 + subject7**

**var percentage = (total / 700) \* 100;**

**//update the result in HTML**

**document.getElementById("total").innerHTML = total + "/700"**

**document.getElementById("percentage").innerHTML = total / 700 \* 100 + "%"**

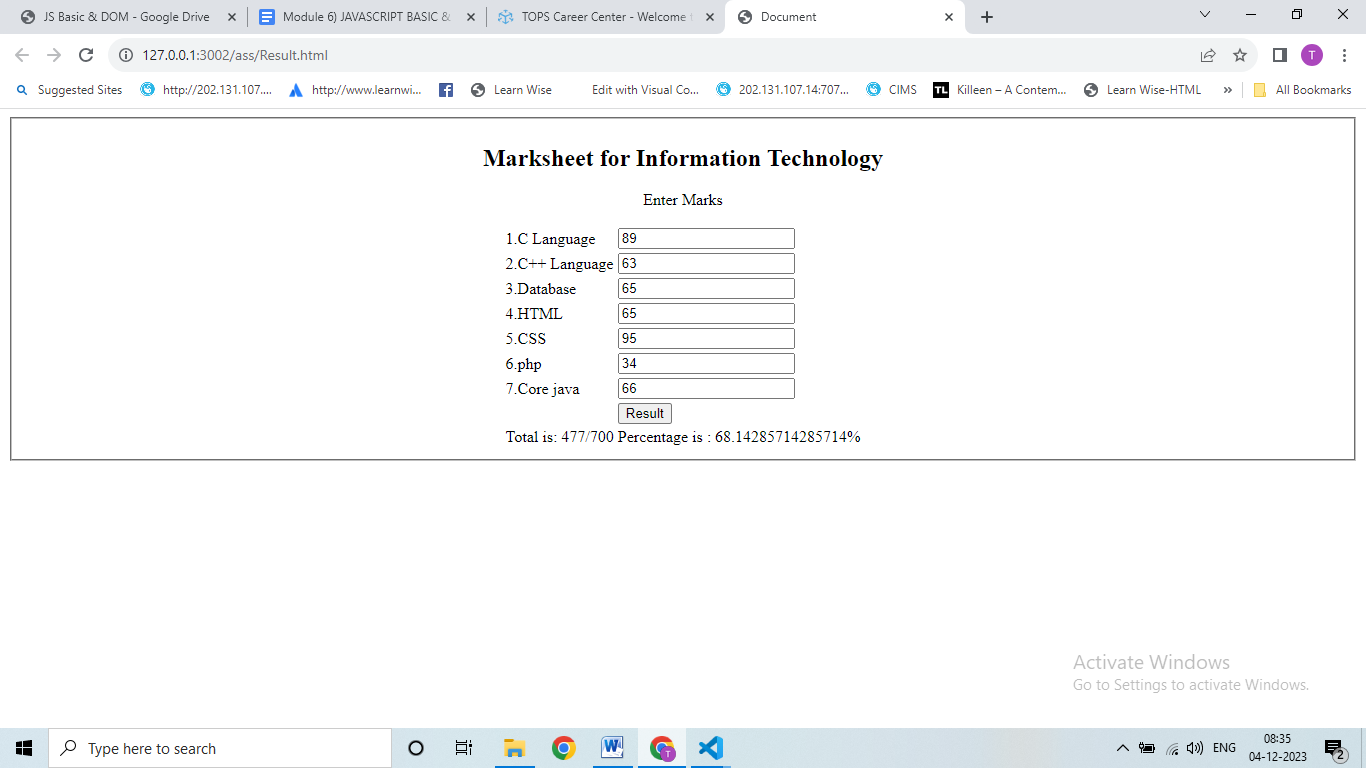
**}**

**</script>**

**</body>**

**</html>**

**OUTPUT:**



**Que-21  Create a slider using JavaScript**

**Ans:**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

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

**<title>Document</title>**

**</head>**

**<style>**

**\* {**

**margin: 0;**

**padding: 0;**

**box-sizing: border-box;**

**}**

**#slider-container {**

**position: relative;**

**max-width: 1500px;**

**margin: auto;**

**overflow: hidden;**

**margin-top: 100px;**

**}**

**#slider {**

**display: flex;**

**transition: transform 0.5s ease-in-out;**

**justify-content: space-around;**

**}**

**.slide {**

**width: 200px;**

**height: 200px;**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**}**

**#prev,**

**#next {**

**position: absolute;**

**top: 50%;**

**transform: translate(-50%);**

**font-size: 20px;**

**cursor: pointer;**

**}**

**#prev {**

**position: relative;**

**margin: 90px 0 0 500px;**

**}**

**#next {**

**position: relative;**

**margin: -27px 0 0 700px;**

**}**

**</style>**

**<body>**

**<div id="slider-container">**

**<div id="slider">**

**<div class="slide" style="background-color: yellow;">Slider 1</div>**

**<div class="slide" style="background-color: red;">Slider 2</div>**

**<div class="slide" style="background-color: rebeccapurple;">Slider 3</div>**

**<div class="slide" style="background-color: green;">Slider 4</div>**

**<div class="slide" style="background-color: blue;">Slider 5</div>**

**</div>**

**<div class="arrow">**

**<div id="prev" onclick="prevSlide()">&#10094;</div>**

**<div id="next" onclick="nextSlide()">&#10095;</div>**

**</div>**

**</div>**

**<script>**

**let currentSlide = 0;**

**const slides = document.querySelectorAll(".slide");**

**const slider = document.getElementById("slider");**

**function showSlide(index) {**

**const newposition = -index \* slides[0].offsetWidth;**

**slider.style.transform = `translateX(${newposition}px)`;**

**}**

**function nextSlide() {**

**currentSlide = (currentSlide + 1) % slides.length;**

**showSlide(currentSlide);**

**updateSlideOrder();**

**}**

**function prevSlide() {**

**currentSlide = (currentSlide - 1 + slides.length) % slides.length;**

**showSlide(currentSlide);**

**updateSlideOrder();**

**}**

**function updateSlideOrder() {**

**const slidesArray = Array.from(slides);**

**const adjustedSlides = slidesArray.slice(currentSlide).concat(slidesArray.slice(0, currentSlide));**

**slider.innerHTML = ""; //clear existing slides**

**adjustedSlides.forEach((slide) => slider.appendChild(slide.cloneNode(true)));**

**}**

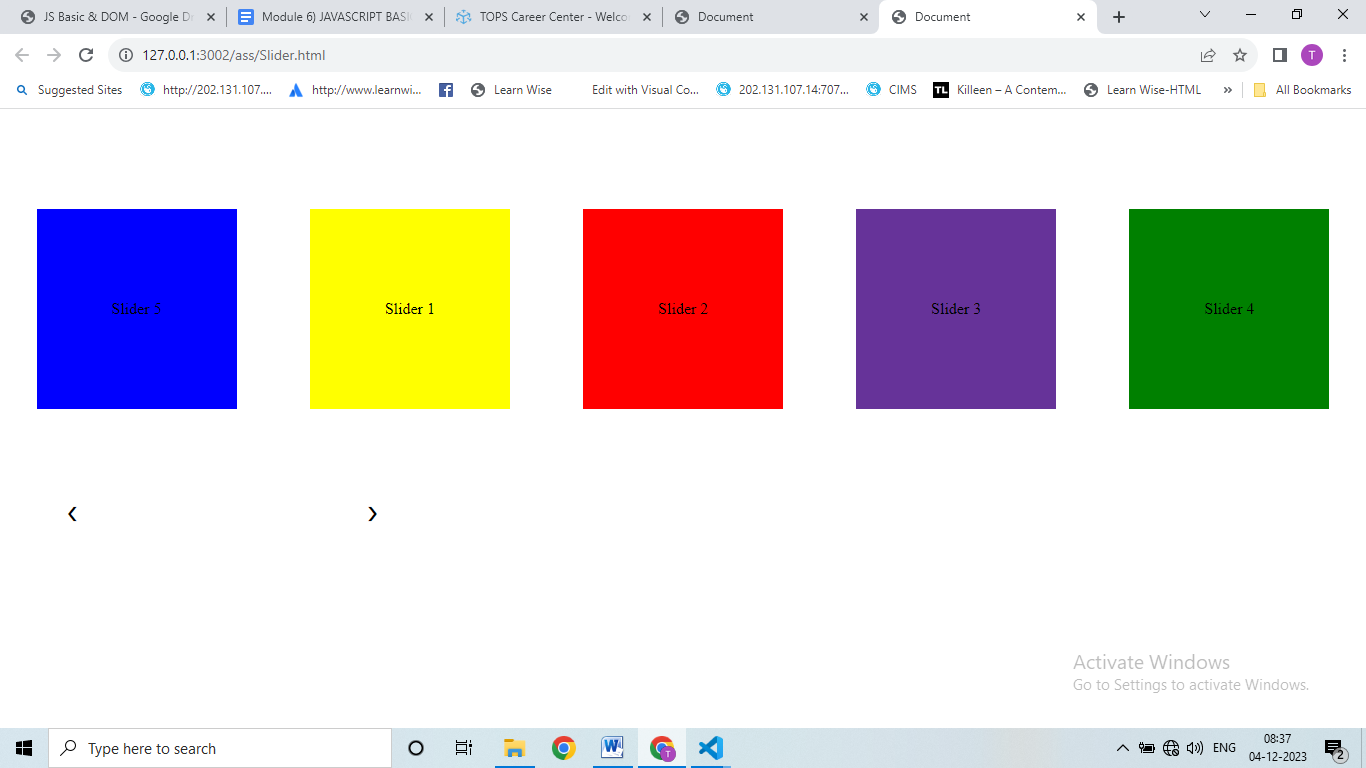
**showSlide(currentSlide);//Initialize the display**

**</script>**

**</body>**

**</html>**

**OUTPUT:**



**Advance JavaScript for Front-End Introduction and Code Quality**

**Que - 1  Write a program to Show an alert**

**Ans:**

        To show an alert in JavaScript, you can use the alert function. Here's a simple program to show an alert:

alert("This is an alert message");

**Que - 2** **What will be the result for these expressions?**

**Ans:**

**1.  5 > 4**

**Ans :** True

**2.  "apple" > "pineapple"**

**Ans :** False

**3.  "2" > "12"**

**Ans:** True (String comparison; "2" is considered greater)

**4.  undefined == null**

**Ans:** True (They are considered equal in loose comparison)

**5.  undefined === null**

**Ans:** False (They are not of the same data type)

**6.  null == "\n0\n"**

**Ans:** False (null is not considered equal to a string)

**7.  null === +"\n0\n"**

**Ans:** False (strict comparison; null and +"\n0\n" are not of the same data type)

**Que - 3  Will alert be shown?**

if ("0") { alert( 'Hello'); }

**Ans:**

      Yes, an alert will be shown because the string "0" is considered truthy, so the condition in the if statement is met.

**Que - 4 What is the code below going to output? alert( null || 2 || undefined );**

**Ans:**

       The code will output 2. It evaluates the logical OR (||) operator, and it returns the first truthy value it encounters, which is 2. If all values are falsy, it returns the last value, which is undefined in this case.

**Que - 5  The following function returns true if the parameter age is greater than 18. Otherwise it asks for a confirmation and returns its result:**

**Function**

**checkAge(age)**

**{**

**if (age> 18) { return true; }**

**else {**

**// ...return confirm (‘did parents allow you?');**

**}**

**}**

**Ans:**

 function checkAge(age) {

  if (age > 18) {

return true;

  } else {

return confirm('Did parents allow you?');

  }

}

**Que-6  Replace Function Expressions with arrow functions in the code below:**

Function

ask(question, yes, no)

{ if (confirm(question))yes();

else

no();

}

ask("Do you agree?", function()

{ alert("You agreed."); },

function() {

alert("You canceled the execution."); }

}

**Ans:**

const ask = (question, yes, no) => {

  if (confirm(question)) {

yes();

  } else {

no();

  }

};

ask(

  "Do you agree?",

  () => {

    alert("You agreed.");

  },

  () => {

    alert("You canceled the execution.");

  }

);

**JavaScript Essentials**

**Que-1   Calculate subtotal price of quantity in JavaScript?**

**Ans:**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

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

**<title>Subtotal Price</title>**

**</head>**

**<style>**

**table {**

**border-collapse: collapse;**

**width: 100%;**

**}**

**th,**

**td {**

**border: 1px solid #ddd;**

**padding: 8px;**

**text-align: left;**

**}**

**th {**

**background-color: #f2f2f2;**

**}**

**.romove-button {**

**background-color: orangered;**

**color: #fff;**

**border: none;**

**padding: 5px 10px;**

**cursor: pointer;**

**}**

**</style>**

**<body>**

**<table>**

**<thead>**

**<th>Product Info</th>**

**<th>Quantity</th>**

**<th>Price per Unit</th>**

**<th>Price Subtotal</th>**

**<th></th>**

**</thead>**

**<tbody>**

**<tr>**

**<td>Product 1</td>**

**<td><input type="number" id="n1" onkeyup="a()"></td>**

**<td>$20</td>**

**<td id="r1">0</td>**

**<td>**

**<button class="romove-button">Remove</button>**

**</td>**

**</tr>**

**<tr>**

**<td>Product 2</td>**

**<td><input type="number" id="n2" onkeyup="a()"></td>**

**<td>$40</td>**

**<td id="r2">0</td>**

**<td>**

**<button class="romove-button">Remove</button>**

**</td>**

**</tr>**

**<tr>**

**<td></td>**

**<td></td>**

**<td></td>**

**<td id="total">0</td>**

**<td></td>**

**</tr>**

**</tbody>**

**</table>**

**<script>**

**function a() {**

**var n1 = document.getElementById("n1").value;**

**var r1 = document.getElementById("r1").innerHTML = "$" + (20 \* n1);**

**var n2 = document.getElementById("n2").value**

**var r2 = document.getElementById("r2").innerHTML = "$" + (40 \* n2);**

**var total = document.getElementById("total").innerHTML = "$" + (20 \* n1 + 40 \* n2)**

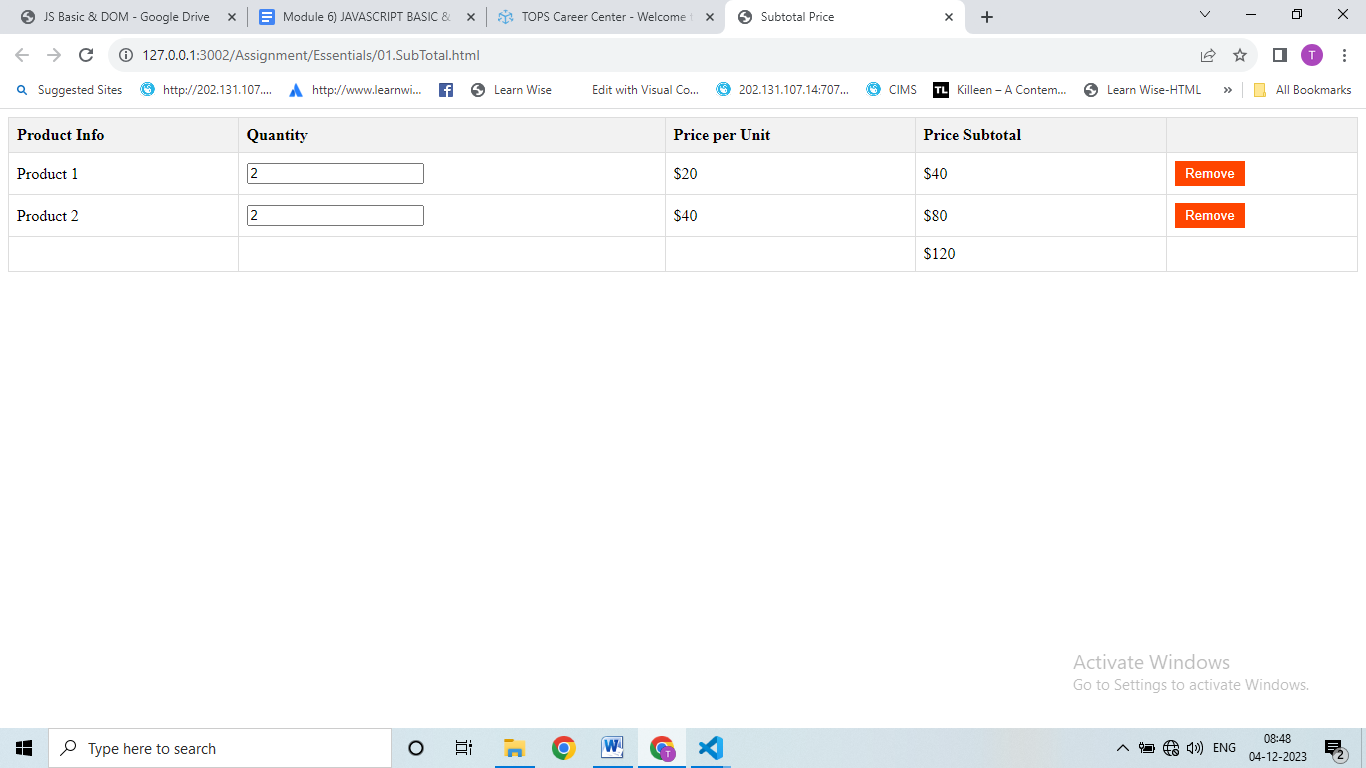
**}**

**</script>**

**</body>**

**</html>**

**OUTPUT:**



**Que - 2**  **What is JavaScript Output method?**

**Ans:**

            JavaScript doesn't have a specific "output method" in the same way some other languages do. However, JavaScript is commonly used to manipulate and interact with the Document Object Model (DOM) of a webpage, and this interaction often involves displaying output to the user. The primary methods for output in JavaScript include:

1.      console.log(): This method is used for logging messages to the browser console. It's commonly used for debugging purposes.

console.log("Hello, world!");

2.      DOM Manipulation: JavaScript can dynamically manipulate the content of an HTML document by interacting with the DOM. For example, changing the text content of an HTML element:

document.getElementById("exampleElement").innerText = "New Text";

or appending HTML content:

document.getElementById("exampleElement").innerHTML = "<p>New HTML content</p>";

**Que- 3   How to used JavaScript Output method?**

**Ans:**

          Using JavaScript to output information typically involves using console.log() for debugging or updating HTML content through the DOM for user-facing output.

Here's an example of using console.log():

console.log("Hello, world!");

And an example of updating HTML content using the DOM:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>JavaScript Output Example</title>

</head>

<body>

<p id="output">Initial content</p>

<script>

     // JavaScript code to update the content of the paragraph

     document.getElementById("output").innerText = "Updated content";

</script>

</body>

</html>

**Que- 4  How to use JavaScript Events to do all examples**

**Ans:**

             JavaScript events are actions or occurrences that happen in the browser, such as a user clicking a button or resizing the window. You can use events to trigger JavaScript code. Here's an example of using JavaScript events:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>JavaScript Events Example</title>

<style>

     button {

         padding: 10px;

         font-size: 16px;

         cursor: pointer;

     }

</style>

</head>

<body>

<button id="myButton">Click me</button>

<script>

     // JavaScript code to handle the click event

        document.getElementById("myButton").addEventListener("click", function() {

         alert("Button clicked!");

     });

</script>

</body>

</html>

**In this example:**

The HTML button element has an ID of "myButton."

The JavaScript code uses addEventListener to listen for a "click" event on the button.

When the button is clicked, the provided function is executed, showing an alert with the message "Button clicked!"

This is a basic example, and JavaScript events can be used for a wide range of interactions, such as handling form submissions, responding to keyboard input, and more.