**“JAVASCRIPT”**

# Q1. what is javascript?

Ans : Javascript also known as JS is a dynamic programming language that used for web application , game development , and lots more. It allows you to implement dynamic features on web pages that cannot be done with only HTML and CSS .

Javascript is used to make webpages interactive and dynamic. For Example:-

Animations ,popup menus ,clickable buttons etc.

It can be used for both client as well as server side of development.

# Q2. What is the use of NaN function?

Ans: Definition and usage :-

* In javascript NaN is a short for “Not-a-Number.
* The NaN () method returns true if a value is NAN.
* The NaN method converts the value to a number before

testing it.

# Q3. What is negative Infinity?

Ans : The negative infinity in JavaScript is a constant value that is used

to represent a value which is the lowest available. This means that no other

number is lesser 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:

1. Negative infinity results in 0 when divided by any other number.
2. When divided by itself or positive infinity, negative infinity return

NaN.

1. Negative infinity, when divided by any positive number (apart from

positive infinity) is negative infinity.

1. If we multiply negative infinity with NaN, we will get NaN as a result.
2. The product of NaN and negative infinity is 0.
3. The product of two negative infinities is always a positive infinity
4. The product of two negative infinities is always a positive infinity.
5. The product of both positive and negative infinity is always negative

infinity.

# Example:

<!DOCTYPE html>

<html>

<head>

<meta charset=”UTF-8”>

<meta Http-equiv=”X-UA-Compatible” content=”IE=edge”>

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

<title>document<title>

</head>

<body>

    <style>

        h1 {

            color: green;

        }

    </style>

    <h1>Tops Technology</h1>

    <h1>

      What is negative infinity in JavaScript?

  </h1>

    <button onclick=“TopsNegativeInfinity()”>

      Generate negative infinite

  </button>

    <p id=”Tops”></p>

    <script>

        function TopsNegativeInfinity() {

            //negative value greater than the

            //largest representable number in JavaScript

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

            document.getElementById(“Tops”).innerHTML = n;

        }

   </script>

</body>

</html>

# Q4 . Which company developed JavaScript ?

Ans : JavaScript is a scripting language developed by Netscape. It can be used to program web browser or even servers. It can dynamically update the contents of the webpage, which is the beauty of JavaScript.

# Q5. What are undeclared and undefined variables?

Ans :

* Undefined variable means a variable has been declare but does not have a value.
* Undeclared variable means that the variable does not

exist in the program at all.

# Undefined Example

var dog;

console.log(dog);

Output:

Undefined

Undeclared Example

console.log(cat);

ReferenceError: cat is not defined

# Q6. Write the code for adding new elements dynamically?

Ans :

Example :

<!DOCTYPE html>

<html>

<head>

<meta charset=”UTF-8”>

<meta Http-equiv=”X-UA-Compatible” content=”IE=edge”>

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

<title>Adding New Elements</title>

</head>

<script type=“text/javascript”>

function addNode() { var newP = document.createElement(“p”);

var textNode = document.createTextNode(“ This is a new text node”);

newP.appendChild(textNode);

document.getElementById(“firstP”).appendChild (newP); }

</script>

</head>

<body> <p id=“firstP”>firstP<p> </body>

</html>

# Q7. What is the difference between ViewState and SessionState?

Ans:

|  |  |
| --- | --- |
| 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 user until the browser is closed or there is session expiration. |
| * Information is stored on the client’ send only. | * Information is stored on the server. |
| * Information is stored on the client’s end only. | * used for the persistence of user specific data on the server end. |
| * ViewState values are lost/clearedwhen new page is loaded. | * SessionState can be cleared by   programmer or user or in case of timeouts. |

# Q8. What is === operator?

Ans. The strict equality operator (===) checks whether its two operands are equal, returning a Boolean result. Unlike the equality operator, the strict equality operator always considers operands of different types to be different.

# Q9. How can the style/class of an element be changed?

Ans.

**1: Changing CSS with the help of the style property:**

**Syntax:**

document.getElementById(“id”).style.property = new\_style

**Example:** In this example, we have built a PAN number validator. First, we

will take the input value and match it with a regex pattern. If it matches then

using JavaScript add an inline style on the <p> tag. Otherwise, add a

different style on the <p> tag.

<!DOCTYPE html>

<**html** lang=“en”>

<**head>**

<meta charset=”UTF-8”>

<meta Http-equiv=”X-UA-Compatible” content=”IE=edge”>

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

<title>document</title>

<**/head**>

<**body**>

    <**h1** style=“color: green;”>

        GeeksforGeeks

    <**/h1**>

    <**h2**>

        How can the style/class of

        an element be changed?

    <**/h2**>

    <**b**>Validate Pan Number<**/b**>

    <**input** type=“text” id=“pan” />

    <**p**><**/p**>

    <**button** id=“submit”>Validate<**/button**>

    <**script**>

**const** btn = document.getElementById(“submit”);

        btn.addEventListener(“click”, function () {

**const** pan = document.getElementById(“pan”).value;

**const** para = document.querySelector(“p”);

            let regex = /([A-Z]){5}([0-9]){4}([A-Z]){1}$/;

**if** (regex.test(pan.toUpperCase())) {

                para.innerHTML = “Hurrey It’s correct”;

                // Inline style

                para.style.color = “green”;

            } **else** {

                para.innerHTML = “OOps It’s wrong!”;

                // Inline style

                para.style.color = “red”;

            }

        });

    <**/script**>

<**/body**>

<**/html**>

**2. The className Property:**This property is used to set the current class of

the element to the specified class.

**Syntax:**

document.getElementById(“id”).className = class

**Example:**

* HTML

<!DOCTYPE html>

<**html** lang=“en”>

**<head**>

<meta charset=”UTF-8”>

<meta Http-equiv=”X-UA-Compatible” content=”IE=edge”>

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

<**title**>document<**/title**>

    <**style**>

        .colorBlue {

            color: blue;

        }

        .colorRed {

            color: red;

        }

    <**/style**>

<**/head**>

<**body**>

    <**h1** style=“color: green;”>

        GeeksforGeeks

    <**/h1**>

    <**h2**>

        How can the style/class of

        an element be changed?

    <**/h2**>

    <**h3**>className Example<**/h3**>

    <**p** class=“colorBlue”>

        GeeksforGeeks is a computer science portal

        for geeks.This platform has been designed

        for every geek wishing to expand their

        knowledge, share their knowledge and is

        ready to grab their dream job. GFG have

        millions of articles, live as well

        as online courses, thousands of tutorials

        and much more just for the geek inside you.

    <**/p**>

    <**button** id=“submit”>Change Color<**/button>**

    <**scrip**t>

**const** btn = document.getElementById(“submit”);

**const** para = document.querySelector(“p”);

        btn.addEventListener(“click”, function () {

            para.className = “colorRed”;

        });

    <**/script**>

<**/body**>

<**/html**>

# Q10. How to read and write a file using JavaScript?

Ans. Reading from the file

After the File System module is imported, the reading of the file in JavaScript can be

done by using the readFile() function.

Syntax

The syntax to read from a file is as follows −

readFile(path, format, callBackFunc)

The readFile() function accepts three parameters including one optional parameter.

* Path − The first parameter is the path of the test file from which the

contents are to read. If the current location or directory is the same

directory where the file which is to be opened and read is located then,

only the file name has to be given.

* Format − The second parameter is the optional parameter which is the format of the text file. The format can be ASCII, utf-8 etc.
* CallBackFunc − The third parameter is the call back function which

takes the error as the parameter and displays the fault is any raised due to the error.

Example:

Following example tries to read the contents of the file populate in the

previous example and print it –

const **fs** = require(“fs”)

fs.readFile(“tp.txt”, (err, inputD) => {

if (err) throw err;

console.log(inputD.**toString**());

})

Output

Following is the output of the above example –

You are reading the content from Tutorials Point . The text which is displayed in the console is the text which is in the given file.

Write operation on a file

After the File System file is imported then, the writeFile() operation is

called. The writeFile() method is used to write into the file in JavaScript.

The syntax of this method is as follows −

writeFile(path ,inputData ,callBackFunction)

The writeFile() function accepts three parameters −

* Path − The first parameter is the path of the file or the name

of the file into which the input data is to be written.

If there is a file already, then the contents in the file are deleted and the input which is given by the user will get updated or if the file is not present, then the file with that will be created in the given path and the input information is written into it.

* inputData − The second parameter is the input data which contains the data to be written in the file that is opened.
* callBackFuntion − The third parameter is the function which is the call back function which takes the error as the parameter and shows the fault if the write operation fails.

Example:

Following is an example of the write operation in files in JavaScript.

Const fs = require(“fs”)

Let fInput = “You are reading the content from Tutorials Point”

fs.writeFile(“tp.txt”, fInput, (err) => {

if (err) throw err;

else{

console.log(“The file is updated with the given data”)

}

})

# 11. What are all the looping structures in JavaScript?

Ans. JavaScript supports different kinds of loops:

* for - loops through a block of code a number of times.
* for/in - loops through the properties of an object.
* for/of - loops through the values of an iterable object.
* while - loops through a block of code while a specified condition is true.

# 12. How can you convert the string of any base to an integer in JavaScript?

Ans. Given a string containing an integer value and along with that user

passes a base value. We need to convert that string of any base value to an

integer in JavaScript.

String Integer

“1002” 1002

For performing the above-illustrated task, we would be using a method (or a

function) provided by JavaScript called as parseInt().

This is a special method, provided by JavaScript, that takes an integer value

(of any base which is either specified or not) and further converts the string

into an integer value.

Syntax:

* Following is the syntax that a user may use to convert a string into

an integer value (of any base) -

parseInt(string\_value , base)

* Alternatively, if we don’t want to specify the base value and just

want to convert our string value into an integer value itself, then we

may use the following syntax also -

parseInt(string\_value)

Default value returned by base or radix of parseInt() method is 10. In other

words, if we don’t specify any base or radix value then it by default converts

the string value to an integer value by taking into regard the base or radix

value as 10.

Let us visualize all of the above-illustrated facts with some of the following

examples-

Example:  In this example, we would be passing the string value in a method

(which is explicitly declared for ease purpose) and further that string value is

passed inside the parseInt() method which then further converts that string

value in the corresponding integer value.

# JavaScript

<script>

    let stringConversion = (string\_value) => {

      console.log(“Initial Type: “ + typeof string\_value);

      let integer\_value = parseInt(string\_value);

      console.log(“Final Type: “ + typeof integer\_value);

      console.log(integer\_value);

    };

    stringConversion(“512000”);

    stringConversion(“126410”);

    stringConversion(“0x8975”);

</script>

Output:

Initial Type: string

Final Type: number

512000

Initial Type: string

Final Type: number

126410

Initial Type: string

Final Type: number

35189

Q13. What is the function of the delete operator?

Ans. The delete operator in JavaScript is used to delete an object’s

property.

If it is used to delete an object property that already exists, it

returns true and removes the property from the object. However, deleting

an object property that doesn’t exist will not affect the object, but will still

return true.

The only time false will be returned is when the delete operator is used to

delete a variable or a function.

Syntax

The syntax for using the delete operator is as follows:



Parameters

object: This is the object whose property we want to delete.

property: This is the property to be deleted.

Return value

The delete operator returns true if the specified property is deleted, or false

if the property is not deleted.

Code

In the code below, an object is created and the delete operator is used to

delete some of its properties:

In the code below, an object is created and the delete operator is used to

delete some of its properties:



Output

true

true

{ name: “John Doe”, age: 15 }

{ name: “Buddy”, age: 2 }

# Q14. 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 in JavaScript namely Alert Box, Confirm Box and Prompt Box.

Alert Box: It is used when a warning message is needed to be produced.

When the alert box is displayed to the user, the user needs to press ok and

proceed.

Syntax:

alert(“this is a alert box”)

Confirm Box: It is a type of pop-up box that is used to get authorization or permission from the user. The user has to press the ok or cancel button to proceed.

Syntax:

confirm(“this is confirm box”)

Example:

Prompt Box: It is a type of pop up box which is used to get the user input for

further use. After entering the required details user have to click “ok” to

proceed next stage else by pressing the “cancel”button user returns the null

value.

Syntax:

Prompt(“this is Prompt box”)

Examples given below:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

    <title>boxes</title>

</head>

<body>

    <button onclick="a()">alert</button>

    <p id="one"></p>

    <!-- alert -->

    <button onclick="b()">confirm</button>

    <p id="two"></p>

    <!-- confirm -->

    <button onclick=" c()">confirm</button>

    <p id="three"></p>

    <!-- prompt -->

    <script>

        function a() {

            alert("not found")

        }

        // alert box

        2.

        function b() {

              var txt;

              if(confirm("do you really want  to exit")){

                txt='ok';

              }

              else{

                txt='no';

              }

              document.getElementById("two").innerHTML=txt;

        }

        // confirm box

        // 3 prompt box

        function c() {

            let text;

            let person = prompt("Please enter your name:", "pahal bhawsar");

            if (person == null || person == "") {

                text = "please enter your name.";

            } else {

                text = "Hello " + person + "! How are you today?";

            }

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

        }

    </script>

</body>

</html>

## 15. What is the use of Void (0)?

Ans.

Using “javascript:void(0);” in anchor tag: Writing “javascript:void(0);” in

anchor tag can prevent the page to reload and JavaScript functions can be

called on single or double clicks easily.

 Example:

<!DOCTYPE html>

<html lang=”en”>

<head>

<meta charset=”UTF-8”>

<meta Http-equiv=”X-UA-Compatible” content=”IE=edge”>

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

<title>JavaScript:void(0)</title>

</head>

<body>

    <center>

        <h1 style=“color:green”>GeeksforGeeks</h1>

        <h3>JavaScript:void(0)</h3>

        <a href=“javascript:void(0);”

           ondblclick=“alert(“Welcome to Geeks for Geeks”)”>

Double click on me </a>

    </center>

</body>

</html>

# 16. How can a page be forced to load another page in JavaScript?

Ans:

Step 1: Create a file named index.html. Add a heading and two buttons to it.

One button forcefully loads a page with a live URL and the other button loads

a local HTML page. In the <script> tag we have two functions, one loads gfg home page, and the second loads a local HTML page using  window.location property.

<!DOCTYPE html>

<html lang=“;en”;>

<head>

    <meta charset=“;UTF-8”;>

    <meta http-equiv=“;X-UA-Compatible”;

        content=“;IE=edge”;>

    <meta name=“;viewport”; content=

        ”;width=device-width, initial-scale=1.0”;>

</head>

<body>

    <h3>This is the original page</h3>

    <br>

    <button onclick=“;force\_load\_gfg()”;>

        Force Load GFG Page

    </button>

    <br><br>

    <button onclick=“;force\_load\_local()”;>

        Force Load Local HTML page

    </button>

    <script>

        function force\_load\_gfg() {

            window.location =

                ”;https://www.geeksforgeeks.org/”;

        }

        function force\_load\_local() {

            window.location =

                ”;F:/gfg/PageRedirect/newPage.html”;

        }

    </script>

</body>

</html>

Step 2: Create a file named newPage.html. This is the local HTML page

that would be loaded by Javascript.

<!DOCTYPE html>

<html lang=en”>

<head>

    <meta charset=“UTF-8”>

    <meta http-equiv=“X-UA-Compatible”

        content=“IE=edge”>

    <meta name=“viewport” content=

        “width=device-width, initial-scale=1.0”>

    <title> New Page </title>

</head>

<body>

    <h3>This is the new loaded page</h3>

</body>

</html>

# 17. What are the disadvantages of using inner HTML in JavaScript?

# Ans: Disadvantages of using innerHTML property in JavaScript :

* The use of innerHTML very slow: The process of using

innerHTML is much slower as its contents as 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.

* Appending to innerHTML is not supported: Usually, += is used

for appending in JavaScript. But on appending to an Html tag using

innerHTML, the whole tag is re-parsed.

Example:

<p id=“geek”>Geeks</p>

title = document.getElementById(“#geek”)

**// The whole “geek” tag is reparsed**

Title.innerHTML += “<p>forGeeks </p>”;

* 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 becomes 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.

* Can also be used for Cross-site Scripting(XSS): The fact that

innerHTML can add text and elements to the webpage, can easily

be used by malicious users to manipulate and display undesirable

or harmful elements within other HTML element tags. Cross-site

Scripting may also lead to loss, leak and change of sensitive

information.

Example:

<!DOCTYPE html>

<html>

<head>

<meta charset=”UTF-8”>

<meta Http-equiv=”X-UA-Compatible” content=”IE=edge”>

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

    <title>

        Using innerHTML in JavaScript

    </title>

</head>

<body style=“text-align: center”>

    <h1 style=“color:green”>

        GeeksforGeeks

    </h1>

    <p id=“P”>

        A computer science

        portal for geeks.

    </p>

    <button onclick=“geek()”>

        Try it

    </button>

     <p id=“p”></p>

     <script>

        function geek() {

            var x = document.getElementById(“P”)  .innerHTML;

            document.getElementById(“p”) .innerHTML = x;

            document.getElementById(“p”) .style.color = “green”;

        }

    </script>

</body>

  </html>

# *Q. Create a Marksheet for Infornation Technology.*

Ans:

HTML & CSS for Markshheet :-

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

    <title>Document</title>

    <style>

        \* {

            margin: 0;

            padding: 0;

            box-sizing: border-box;

        }

        body {

            width: 100%;

            text-transform: capitalize;

            font-family: Arial, Helvetica, sans-serif;

        }

        form {

            margin: auto;

            padding: 1rem 2rem;

            width: 30rem;

            border: 2px solid #030303;

        }

        h2 {

            margin: 1rem 0;

            text-align: center;

        }

        input {

            width: 10rem;

            margin: .5rem 4rem;

            font-size: bold;

            padding: .2rem;

            border: 2px solid #000;

        }

        label {

            font-weight: bolder;

        }

        button {

            padding: .5rem 1rem;

            font-weight: bold;

            font-size: 1.2rem;

            background: #b0402a;

            border: 3px solid #000;

            outline: 3px solid #b0402a;

            outline-offset: -.5rem;

            transition: .2s ease;

            cursor: pointer;

        }

        button:hover {

            outline: 3px solid #000000;

            outline-offset: .1rem;

        }

        #display{

            margin: 1rem;

            display: flex;

            justify-content: space-around;

            align-items: center;

        }

       #display #total{

        font-size: 1.2rem;

        font-weight: bold;

       }

       #display #percent{

        font-size: 1.2rem;

        font-weight: bold;

       }

       #display #grade{

        font-size: 1.2rem;

        font-weight: bold;

       }

    </style>

</head>

<body>

    <form action="" onsubmit="return result()">

        <h2>Marks </h2>

        <table>

            <caption>

                <h3>enter marks</h3>

            </caption>

            <tr>

                <td><label for="sub1">c language</label></td>

                <td><input type="number" name="sub1" id="sub1" required min="0" max="100"

                        placeholder="Enter Your Marks"></td>

            </tr>

            <!-- 1 -->

            <tr>

                <td><label for="sub1">c++</label></td>

                <td><input type="number" name="sub2" id="sub2" required min="0" max="100"

                        placeholder="Enter Your Marks"></td>

            </tr>

            <!-- 2 -->

            <tr>

                <td><label for="sub1">database</label></td>

                <td><input type="number" name="sub3" id="sub3" required min="0" max="100"

                        placeholder="Enter Your Marks"></td>

            </tr>

            <!-- 3 -->

            <tr>

                <td><label for="sub1">HTML</label></td>

                <td><input type="number" name="sub4" id="sub4" required min="0" max="100"

                        placeholder="Enter Your Marks"></td>

            </tr>

            <!-- 4 -->

            <tr>

                <td><label for="sub1">CSS</label></td>

                <td><input type="number" name="sub5" id="sub5" required min="0" max="100"

                        placeholder="Enter Your Marks"></td>

            </tr>

            <!-- 5 -->

            <tr>

                <td><label for="sub1">core java</label></td>

                <td><input type="number" name="sub6" id="sub6" required min="0" max="100"

                        placeholder="Enter Your Marks"></td>

            </tr>

            <!-- 6 -->

            <tr>

                <td><label for="sub1">php</label></td>

                <td><input type="number" name="sub7" id="sub7" required min="0" max="100"

                        placeholder="Enter Your Marks"></td>

            </tr>

            <!-- 7 -->

            <tr>

                <td></td>

                <td>

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

                </td>

            </tr>

        </table>

        <div id="display">

            <div id="total">Total:</div>

            <div id="percent">Percent:</div>

            <div id="grade">Grades:</div>

        </div>

    </form>

    <script src="marksheet.js"></script>

</body>

</html>

# *Javascript for Marksheet :-*

function result() {

    // firstly make function and name it

    let sub1 = document.getElementById('sub1').value;

    let sub2 = document.getElementById('sub2').value;

    let sub3 = document.getElementById('sub3').value;

    let sub4 = document.getElementById('sub4').value;

    let sub5 = document.getElementById('sub5').value;

    let sub6 = document.getElementById('sub6').value;

    let sub7 = document.getElementById('sub7').value;

    // declare valriable for all subjects and get their "value" by 'id'

    let total;

    total = +sub1 + +sub2 + +sub3 + +sub4 + +sub5 + +sub6 + +sub7;

    // declare a variable "total" and then store all the subjects addition in the variable.

    document.getElementById('total').innerHTML = total;

    // for printing the "total"

    let percent;

    percent = Math.round((total / 700) \* 100);

    document.getElementById('percent').innerHTML = percent + "%";

    // to caculatepercentage

    if (percent >= 80) {

        document.getElementById('grade').innerHTML = "A"

    } else if (percent >= 65) {

        document.getElementById('grade').innerHTML = "b"

    }

    else if (percent >= 55) {

        document.getElementById('grade').innerHTML = "c"

    }

    else if (percent >= 40) {

        document.getElementById('grade').innerHTML = "d"

    }

    else if (percent >= 33) {

        document.getElementById('grade').innerHTML = "e"

    }

    else{

        document.getElementById('grade').innerText = "fail"

    }

    return false;

    // return false dena padta hai

}

# Q. create Maths operation.

Ans:-

# HTML and CSS for Maths Operation:-

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

    <title>Maths operation</title>

    <style>

        \* {

            margin: 0;

            padding: 0;

            box-sizing: border-box;

            text-transform: capitalize;

            font-weight: bold;

        }

        body {

            width: 100%;

            font-family: Arial, Helvetica, sans-serif;

        }

        table {

            border: 2px solid #000;

            margin: auto;

            padding: 1rem;

        }

        input {

            margin: 0.5rem;

            padding: 0.2rem;

            border: 2px solid #000;

        }

        span {

            margin: 0.5rem;

            padding: 0.2rem;

        }

        button {

            padding: 0.2rem 1rem;

            font-size: 1.2rem;

        }

    </style>

</head>

<body>

    <table>

        <caption>

            <h2>maths operation</h2>

        </caption>

        <tr>

            <td><label for="fnum">enter first number</label></td>

            <td><input type="number" name="fnum" id="fnum" required min="0" onfocus="color()"></td>

        </tr>

        <tr>

            <td><label for="snum">enter second number</label></td>

            <td><input type="number" name="snum" id="snum" required min="0" onfocus="colors()"></td>

        </tr>

        <tr>

            <td>

                <button id="add" onclick="addop()">+</button>

                <button id="subs" onclick="subsop()">-</button>

                <button id="div" onclick="divop()">/</button>

                <button id="times" onclick="timesop()">\*</button>

            </td>

            <td>

                <span>result:</span>

                <span id="result"></span>

            </td>

        </tr>

    </table>

    <script src="mathsop.js">

    </script>

</body>

</html>

# Javascript for Maths Operation:-

function addop() {

    let fnum = document.getElementById('fnum').value;

    let snum = document.getElementById('snum').value;

    let sum = document.getElementById("result").innerHTML = +fnum + +snum;

}

// function for adding the numbers

function subsop() {

    let fnum = document.getElementById('fnum').value;

    let snum = document.getElementById('snum').value;

    let sum = document.getElementById("result").innerHTML = fnum - snum;

}

// function for substracting the numbers

function divop() {

    let fnum = document.getElementById('fnum').value;

    let snum = document.getElementById('snum').value;

    let sum = document.getElementById("result").innerHTML = fnum / snum;

}

// function for division of the numbers

function timesop() {

    let fnum = document.getElementById('fnum').value;

    let snum = document.getElementById('snum').value;

    let sum = document.getElementById("result").innerHTML = fnum \* snum;

}

// function for multiplying the numbers

function color() {

    document.getElementById("fnum").style.backgroundColor = '#0ca8c0ad';

    document.getElementById("fnum").style.color = ' #fff';

}

function colors() {

    document.getElementById("snum").style.backgroundColor = '#0ca8c0ad';

    document.getElementById("snum").style.color = ' #fff';

}

// the above two funtions are for styling of input box

# Q.Create a slider using javacript.

Ans:-

# HTML , CSS and Javascript for slider:-

<!DOCTYPE html>

<html>

<head>

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <style>

        \* {

            box-sizing: border-box

        }

        body {

            font-family: Verdana, sans-serif;

            margin: 0

        }

        .mySlides {

            display: none

        }

        img {

            vertical-align: middle;

        }

        /\* Slideshow container \*/

        .slideshow-container {

            max-width: 1000px;

            position: relative;

            margin: auto;

        }

        /\* Next & previous buttons \*/

        .prev,

        .next {

            cursor: pointer;

            position: absolute;

            top: 50%;

            width: auto;

            padding: 16px;

            margin-top: -22px;

            color: white;

            font-weight: bold;

            font-size: 18px;

            transition: 0.6s ease;

            border-radius: 0 3px 3px 0;

            user-select: none;

        }

        /\* Position the "next button" to the right \*/

        .next {

            right: 0;

            border-radius: 3px 0 0 3px;

        }

        /\* On hover, add a black background color with a little bit see-through \*/

        .prev:hover,

        .next:hover {

            background-color: rgba(0, 0, 0, 0.8);

        }

        /\* Caption text \*/

        .text {

            color: #f2f2f2;

            font-size: 15px;

            padding: 8px 12px;

            position: absolute;

            bottom: 8px;

            width: 100%;

            text-align: center;

        }

        /\* Number text (1/3 etc) \*/

        .numbertext {

            color: #f2f2f2;

            font-size: 12px;

            padding: 8px 12px;

            position: absolute;

            top: 0;

        }

        /\* The dots/bullets/indicators \*/

        .dot {

            cursor: pointer;

            height: 15px;

            width: 15px;

            margin: 0 2px;

            background-color: #bbb;

            border-radius: 50%;

            display: inline-block;

            transition: background-color 0.6s ease;

        }

        .active,

        .dot:hover {

            background-color: #717171;

        }

        /\* Fading animation \*/

        .fade {

            animation-name: fade;

            animation-duration: 1.5s;

        }

        @keyframes fade {

            from {

                opacity: .4

            }

            to {

                opacity: 1

            }

        }

        /\* On smaller screens, decrease text size \*/

        @media only screen and (max-width: 300px) {

            .prev,

            .next,

            .text {

                font-size: 11px

            }

        }

    </style>

</head>

<body>

    <div class="slideshow-container">

        <div class="mySlides fade">

            <div class="numbertext">1 / 3</div>

            <img src="img\_nature\_wide.jpg" style="width:100%">

            <div class="text">Caption Text</div>

        </div>

        <div class="mySlides fade">

            <div class="numbertext">2 / 3</div>

            <img src="img\_snow\_wide.jpg" style="width:100%">

            <div class="text">Caption Two</div>

        </div>

        <div class="mySlides fade">

            <div class="numbertext">3 / 3</div>

            <img src="img\_mountains\_wide.jpg" style="width:100%">

            <div class="text">Caption Three</div>

        </div>

        <a class="prev" onclick="plusSlides(-1)">❮</a>

        <a class="next" onclick="plusSlides(1)">❯</a>

    </div>

    <br>

    <div style="text-align:center">

        <span class="dot" onclick="currentSlide(1)"></span>

        <span class="dot" onclick="currentSlide(2)"></span>

        <span class="dot" onclick="currentSlide(3)"></span>

    </div>

    <script>

        let slideIndex = 1;

        showSlides(slideIndex);

        function plusSlides(n) {

            showSlides(slideIndex += n);

        }

        function currentSlide(n) {

            showSlides(slideIndex = n);

        }

        function showSlides(n) {

            let i;

            let slides = document.getElementsByClassName("mySlides");

            let dots = document.getElementsByClassName("dot");

            if (n > slides.length) { slideIndex = 1 }

            if (n < 1) { slideIndex = slides.length }

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

                slides[i].style.display = "none";

            }

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

                dots[i].className = dots[i].className.replace(" active", "");

            }

            slides[slideIndex - 1].style.display = "block";

            dots[slideIndex - 1].className += " active";

        }

    </script>

</body>

</html>

**“End’s Here”**