**UCCD2323 FRONT-END WEB DEVELOPMENT**

**PRACTICAL 6**

JavaScript Math Object

The JavaScript Math object allows you to perform mathematical tasks on numbers. The math object provides you properties and methods for mathematical constants and functions. Unlike other global objects, Math is not a constructor. All the properties and methods of Math are static and can be called by using Math as an object without creating it.

A white background with black text

Description automatically generated

Math Object Methods

A screenshot of a computer

Description automatically generated

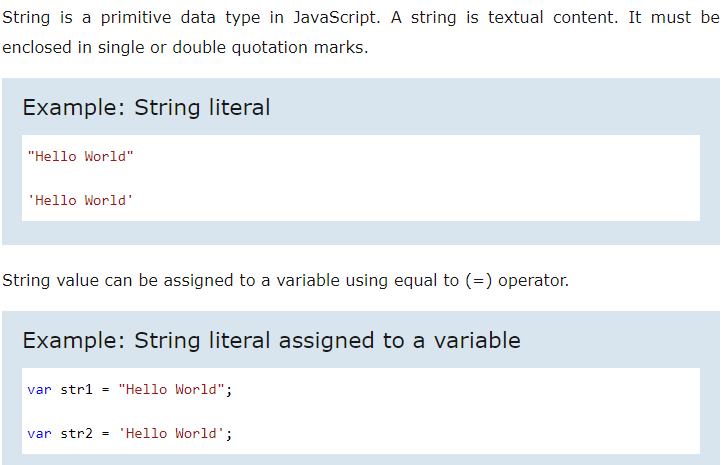
A screenshot of a math test

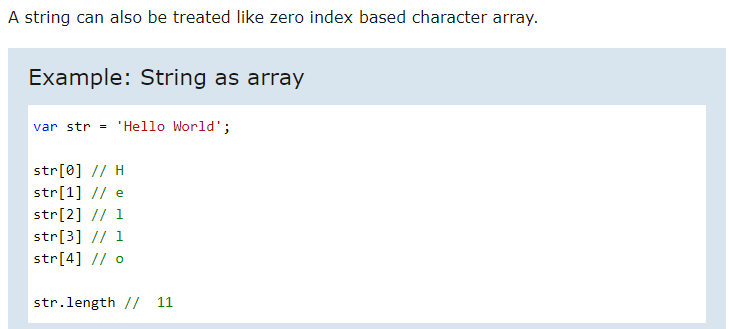
Description automatically generated

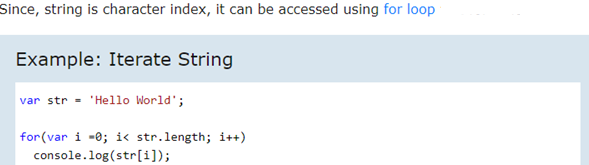
<https://www.w3schools.com/js/js_math.asp>

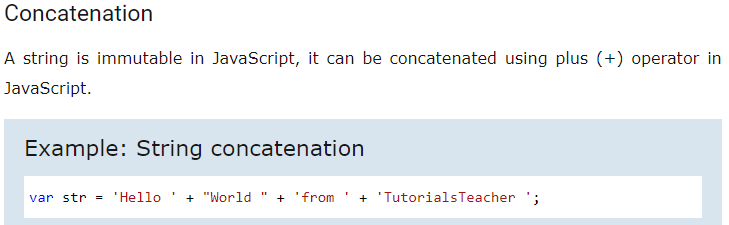
<https://www.w3schools.com/js/js_random.asp>

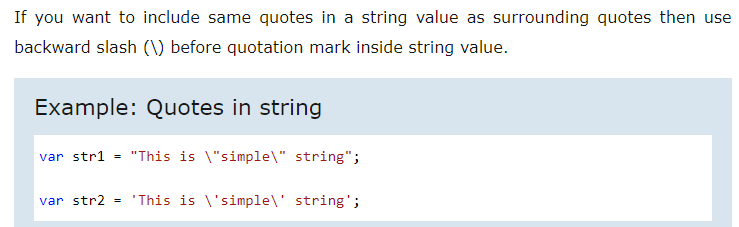
**JavaScript – String**





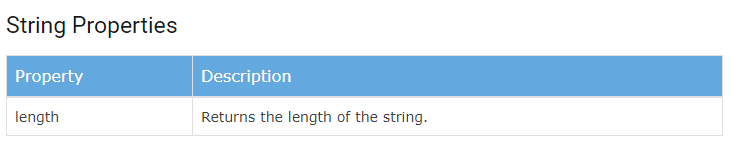


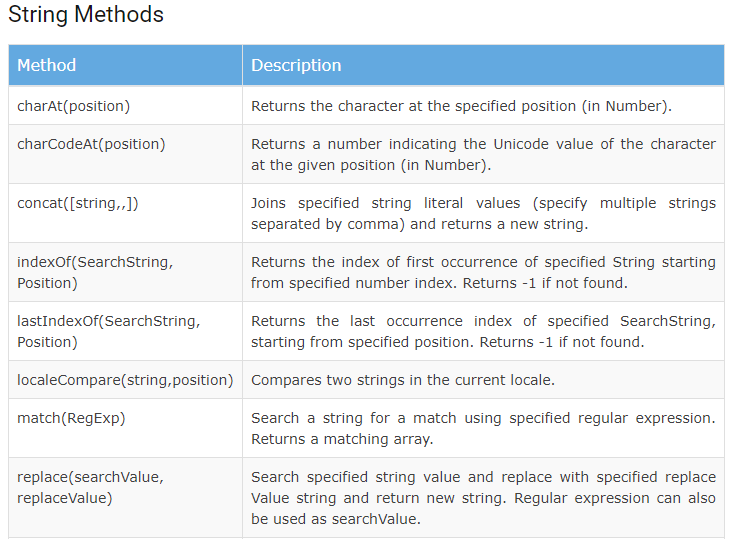


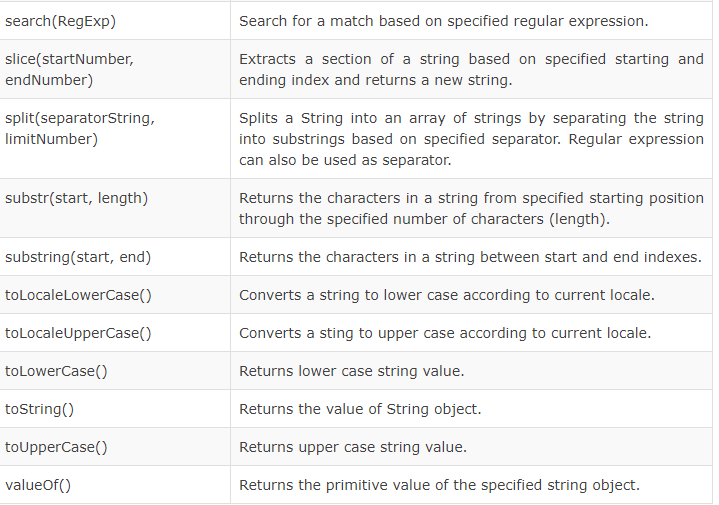


JavaScript String Methods & Properties

JavaScript string (primitive or String object) includes default properties and methods which you can use for different purposes.



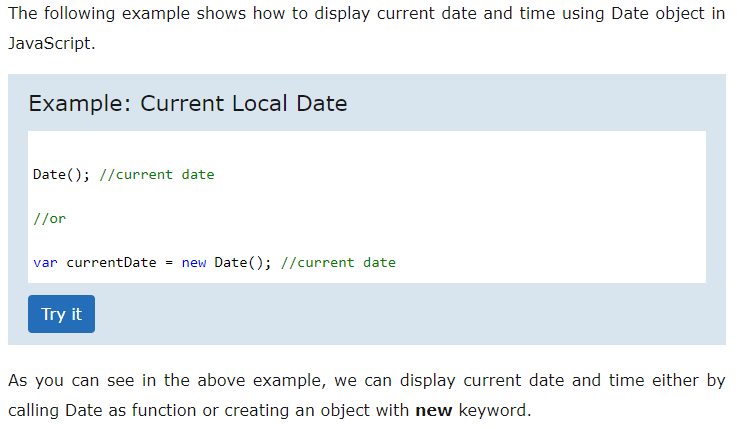


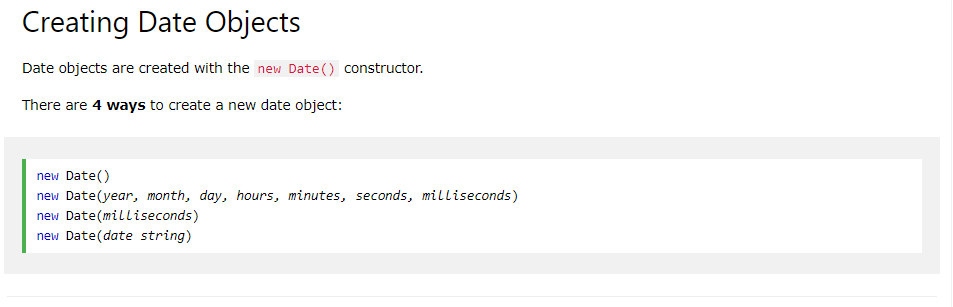


<https://www.w3schools.com/js/js_string_methods.asp>

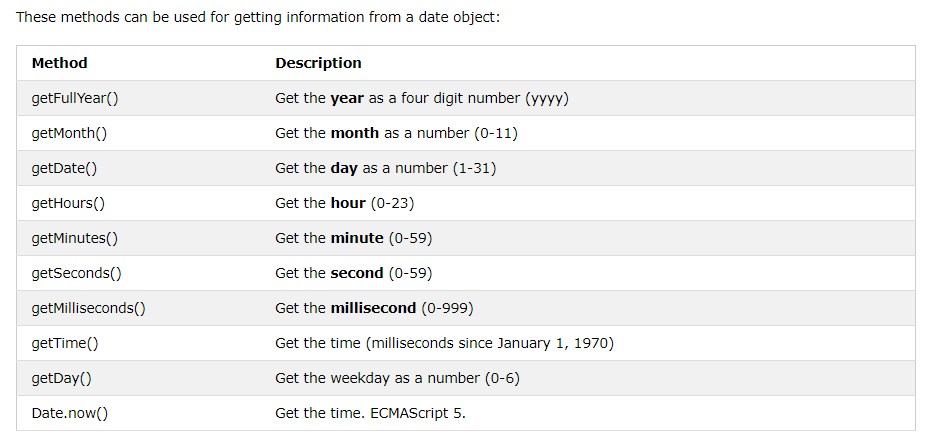
**JavaScript -Date**

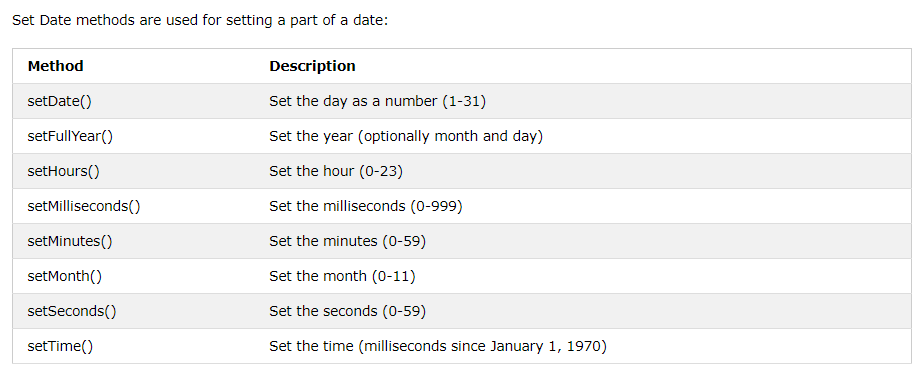
JavaScript provides Date object to work with date & time including days, months, years, hours, minutes, seconds and milliseconds.

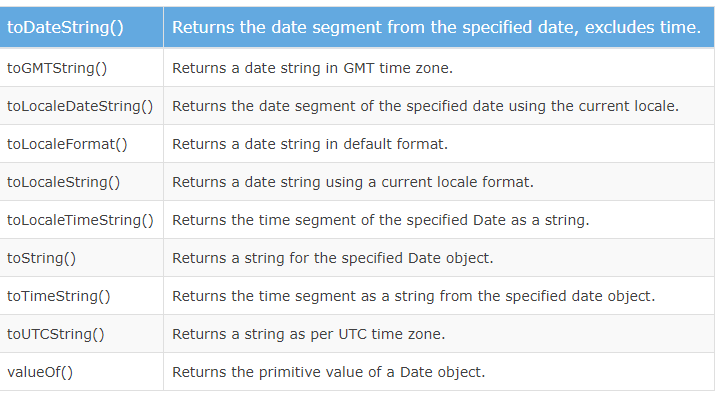




JavaScript Get Date Methods



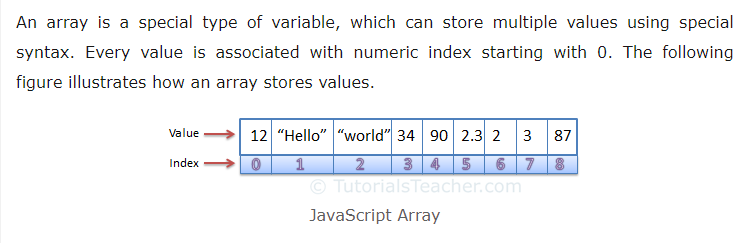




<https://www.tutorialsteacher.com/javascript/javascript-date>

<https://www.w3schools.com/js/js_dates.asp>

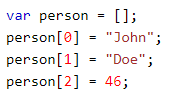
JavaScript Array



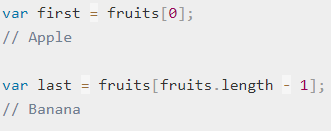
Arrays are list-like objects whose prototype has methods to perform traversal and mutation operations. Neither the length of a **JavaScript array nor the types of its elements are fixed**. Since an array's length can change at any time, and data can be stored at non-contiguous locations in the array, JavaScript arrays are not guaranteed to be dense; this depends on how the programmer chooses to use them.

Create an Array





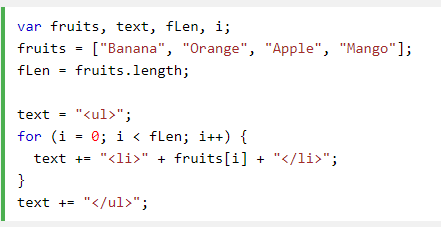
Access (index into) an Array item



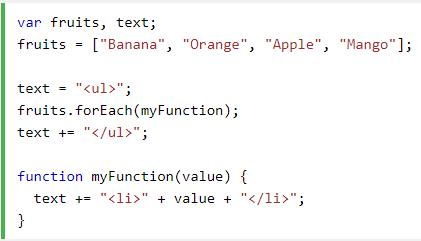


Looping Array Elements

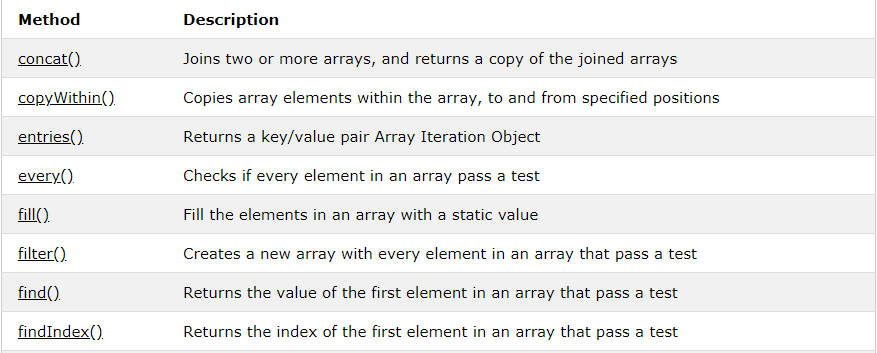
The safest way to loop through an array, is using a for loop:

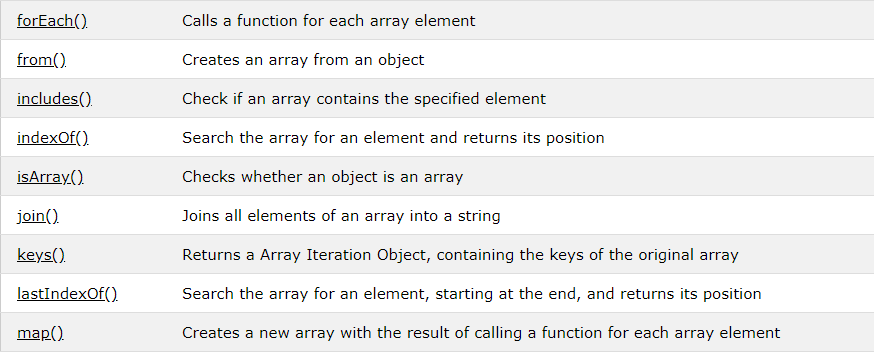


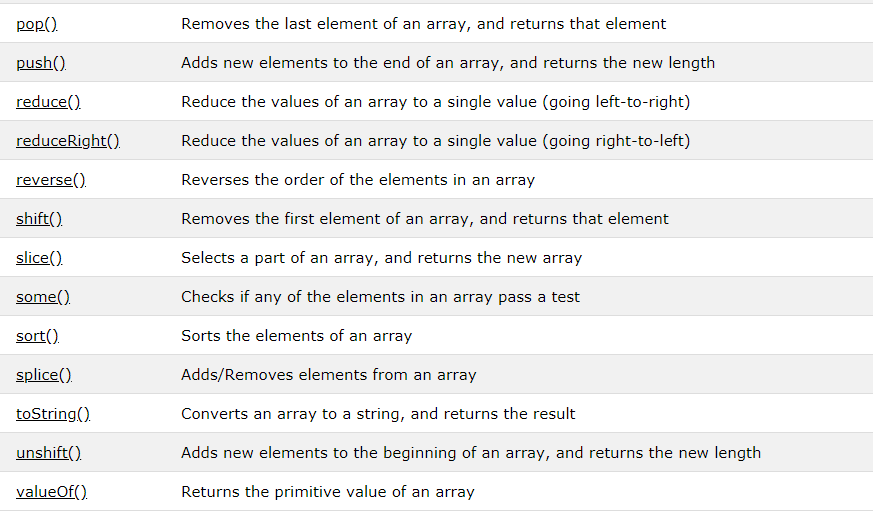
You can also use the Array.forEach() function:



Array Methods









<https://www.w3schools.com/js/js_arrays.asp>

<https://www.w3schools.com/jsref/jsref_obj_array.asp>

JavaScript Object

Object is a non-primitive data type in JavaScript. It is like any other variable, the only difference is that an object holds multiple values in terms of properties and methods. Properties can hold values of primitive data types and methods are functions.

JavaScript objects and JSON objects are different.

In other programming languages like Java or C#, you need a class to create an object of it. In JavaScript, an object is a standalone entity because there is no class in JavaScript. However, you can achieve class like functionality using functions. We will learn how to treat a function as a class in the advance JavaScript section.

In JavaScript, an object can be created in two ways:

* Object literal
* Object constructor

A screenshot of a computer code

Description automatically generated

A green and grey rectangular object with red text

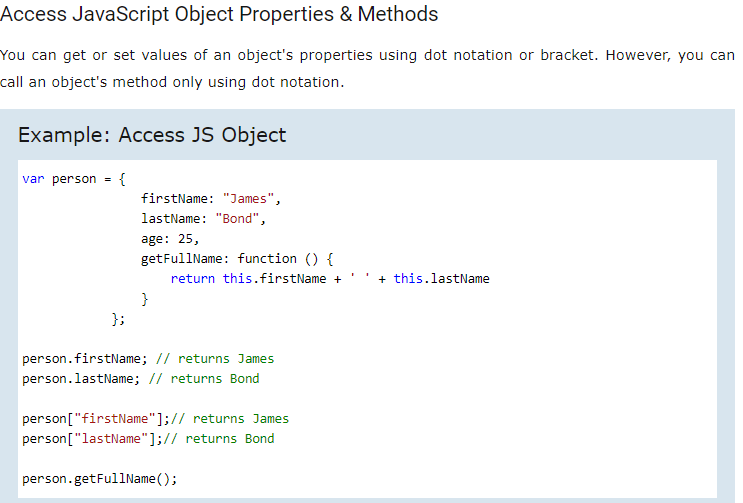
Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated



A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

<https://www.tutorialsteacher.com/javascript/javascript-object>

<https://www.w3schools.com/js/js_object_definition.asp>

**Exercise 1**

1. Create a form to allow user to enter string values. Display all the string values in ascending order and calculate the average number of characters in two decimals for the string values.

[Note: Use “**Inline event handlers**”]

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<title> HTML DOM,Event and Object </title>

</head>

<body>

<form **onsubmit**="addArrayFunction(event)">

<table>

<tr>

<td>

Enter string value

</td>

<td>

:

</td>

<td>

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

</td>

</tr>

<tr>

<td></td>

<td></td>

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

</tr>

</table>

</form>

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

<p id="ave"></p>

<script>

var arr = [];

var total = 0;

function addArrayFunction(event) {

event.preventDefault();

var str = document.getElementById("str").value;

total += str.length;

arr.push(str);

document.getElementById("str").value = "";

arr.sort();

document.getElementById("demo").innerHTML = arr.join('<br>');

document.getElementById("ave").innerHTML = "Average number of characters : " + (total / arr.length).toFixed(2);

}

</script>

</body>

</html>

Create a new empty array

var arr = [];

var str = document.getElementById("str").value;

The getElementById() call returns the input element object with ID ‘str’ . Once we have the object, we can get the value attribute.

document.getElementById("str").value = "";

Set the value attribute to empty

String Length (str.length)

The length property returns the length of a string.

Pushing

The **push()** method adds a new element to an array (at the end).

Sorting an Array

The **sort()** method sorts the elements of an array in place and returns the sorted array. The default sort order is ascending, built upon converting the elements into strings, then comparing their sequences of UTF-16 code units values (https://asecuritysite.com/coding/asc2).

The **join()** method also joins all array elements into a string. It behaves just like toString(), but in addition you can specify the separator such as <br>.

1. Create a form to input from the user a line of text and display the words in alphabetical   
    reverse order in case insensitive JavaScript. Example if you enter “I am good thanks”. The browser will display

thanks

I

good

am

[Note: **Assign events using the HTML DOM**.]

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="utf-8">

    <title> HTML DOM,Event and Object</title>

</head>

<body>

    <form>

        <table>

            <tr>

                <td>

                    Enter a sentence

                </td>

                <td>

                    :

                </td>

                <td>

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

                </td>

            </tr>

            <tr>

                <td></td>

                <td></td>

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

            </tr>

        </table>

    </form>

    <p id="demo"></p>

    <script>

        document.querySelector("form").onsubmit = addArrayFunction;

        function addArrayFunction(event) {

            event.preventDefault();

            var str = document.getElementById("str").value;

            var words = str.split(" ");

            words.sort(function (a, b) {

                if (a.toLowerCase() < b.toLowerCase()) {

                    return -1;

                } else if (a.toLowerCase() > b.toLowerCase()) {

                    return 1;

                } else {

                    return 0;

                }

            });

            words.reverse();

            document.getElementById("demo").innerHTML = words.join('<br>');

            document.getElementById("str").value = "";

        }

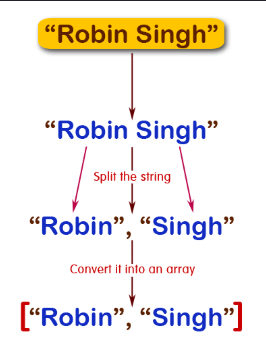
    </script>

</body>

</html>

JavaScript String **split()** Method

The split() method is used to split a string into an array of substrings, and returns the new array. Tip: If an empty string ("") is used as the separator, the string is split between each character. Note: The split() method does not change the original string.

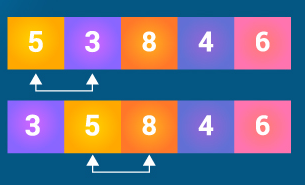


Sorting an Array

The **sort()** method sorts the elements of an array in place and returns the sorted array. The default sort order is ascending, built upon converting the elements into strings, then comparing their sequences of UTF-16 code units values (<https://asecuritysite.com/coding/asc2>). All of the values that start with uppercase letters are sorted before any of the entries that begin with lowercase letters.

How to set up a case-insensitive sort

In order to accomplish the case-insensitive sort, we compare the lowercase versions of the two arguments (a and b). A function passed to the sort method should include two arguments to represent adjacent elements in the array to be sorted. If the first argument is to be sorted before the second, return -1 from the function. If the second argument is to be sorted before the first, return 1. If the sort order doesn't matter, or the two values are equal, return 0.



JavaScript Array **reverse()** method

The JavaScript array reverse() method changes the sequence of elements of the given array and returns the reverse sequence. In other words, the arrays last element becomes first and the first element becomes the last. This method also made the changes in the original array.

1. Create a form to allow user to enter two date and then to compare dates (i.e. greater than, less than or equal to).

[Note: Use “**addEventListener( ) method**”.]

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="utf-8">

    <title>HTML DOM,Event and Object</title>

</head>

<body>

    <form id="myForm">

        <table>

            <tr>

                <td>

                    Enter Start Date

                </td>

                <td>

                    :

                </td>

                <td>

                    <input type="date" id="start" required>

                </td>

            </tr>

            <tr>

                <td>

                    Enter End Date

                </td>

                <td>

                    :

                </td>

                <td>

                    <input type="date" id="end" required>

                </td>

            </tr>

            <tr>

                <td></td>

                <td></td>

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

            </tr>

        </table>

    </form>

    <p id="demo"></p>

    <script>

        document.getElementById("myForm").**addEventListener**("submit", dateCompareFunction);

        function dateCompareFunction(event) {

            event.preventDefault();

            var start = new Date(document.getElementById("start").value);

            var end = new Date(document.getElementById("end").value);

            if (start > end)

                document.getElementById("demo").innerHTML = "start date (" + start + ") is greater than end date (" + end + ")";

            else if (start < end)

                document.getElementById("demo").innerHTML = "start date (" + start + " ) is less than end date (" + end + ")";

            else

                document.getElementById("demo").innerHTML = "start date (" + start + ") is equal to end date (" + end + ")";

            document.getElementById("start").value = "";

            document.getElementById("end").value = "";

        }

    </script>

</body>

</html>

The date object allows us to perform comparisons using the >, <, =, or >= comparison operators, but not the equality comparison operators like ==, !=, ===, and !== (unless we attach date methods to the date Object).

To handle equality comparison, we use the date object alongside the getTime() date method which returns the number of milliseconds. But if we want to compare specific information like day, month, and so on, we can use other date methods like the getDate(), getHours(), getDay(), getMonth() and getYear().

A screenshot of a computer code

Description automatically generated

A screenshot of a computer code

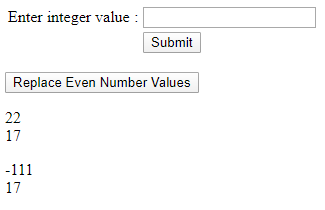
Description automatically generated

Reference:

<https://www.freecodecamp.org/news/javascript-date-comparison-how-to-compare-dates-in-js/>

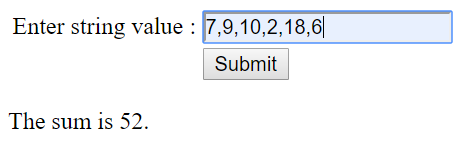
**Exercise 2**

1. Create a form to allow user to enter integer values and store all the values entered in an array. The program then should replace all the even number values found in the array with the value -111 when click “Replace Even Number Values”. Then, display the changes of the array before the replacement of -111 and after the replacement of -111.



Sample Output

1. Create a form to allow user to enter a string containing series of numbers separated by commas. Here is an example of valid input:   
    7,9,10,2,18,6  
   The program should calculate and display the sum of all the numbers.

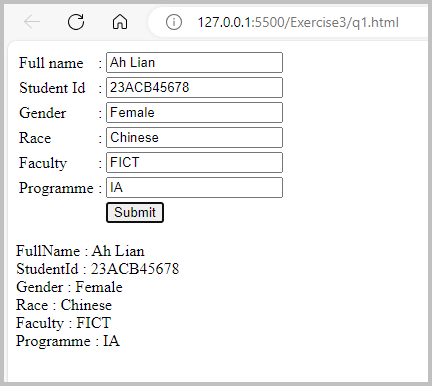


Sample Output

**Exercise 3**

1. Write the program to create a web page with a form that allows a student to enter his/her full name, student ID, gender, race, faculty, and degree program. The program should store this information in a JavaScript object before displaying them in the web page.

Sample output:



Answer:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="utf-8">

    <title>HTML DOM,Event and Object</title>

</head>

<body>

    <form onsubmit="addFunction(event)">

        <table>

            <tr>

                <td>

                    Full name

                </td>

                <td>

                    :

                </td>

                <td>

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

                </td>

            </tr>

            <tr>

                <td>

                    Student Id

                </td>

                <td>

                    :

                </td>

                <td>

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

                </td>

            </tr>

            <tr>

                <td>

                    Gender

                </td>

                <td>

                    :

                </td>

                <td>

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

                </td>

            </tr>

            <tr>

                <td>

                    Race

                </td>

                <td>

                    :

                </td>

                <td>

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

                </td>

            </tr>

            <tr>

                <td>

                    Faculty

                </td>

                <td>

                    :

                </td>

                <td>

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

                </td>

            </tr>

            <tr>

                <td>

                    Programme

                </td>

                <td>

                    :

                </td>

                <td>

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

                </td>

            </tr>

            <tr>

                <td></td>

                <td></td>

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

            </tr>

        </table>

    </form>

    <p id="demo"></p>

 <script>

            function addFunction(event) {

                event.preventDefault();

                var fName = document.getElementById("fName").value;

                var sId = document.getElementById("sId").value;

                var gender = document.getElementById("gender").value;

                var race = document.getElementById("race").value;

                var faculty = document.getElementById("faculty").value;

                var programme = document.getElementById("programme").value;

                var student = { FullName: fName, StudentId: sId, Gender: gender, Race: race, Faculty: faculty, Programme: programme };

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

                for (var key in student) {

                    document.getElementById("demo").innerHTML += key + " : " + student[key] +"<br>";

                }

            }

        </script>

</body>

</html>

var student = { FullName: fName,….}

Create object with properties using object literal syntax.

A screenshot of a computer code

Description automatically generated

1. A parking garage charges a $2.00 minimum fee to park for up to three hours. The garage charges an additional $0.50 per hour for each hour or part thereof in excess of three hours. The maximum charge is $10.00 per day. Assume that no car parks for longer than 24 hours at a time. Write the code to create a web page that has a form to allow the user to input the car plate number and the number of hours parked for each customer. The web page has a function called calculateCharges that calculates the parking charges for parking the car in the garage. The program should use the function calculateCharges to determine the charge for each customer and display all the customer parking information with parking hours displayed in descending order.

Sample Output:

A screenshot of a computer

Description automatically generated

Answer:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="utf-8">

    <title>HTML DOM,Event and Object</title>

</head>

<body>

    <form onsubmit="addFunction(event)">

        <table>

            <tr>

                <td>

                    Car Plate

                </td>

                <td>

                    :

                </td>

                <td>

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

                </td>

            </tr>

            <tr>

                <td>

                    Parking Hours

                </td>

                <td>

                    :

                </td>

                <td>

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

                </td>

            </tr>

            <tr>

                <td></td>

                <td></td>

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

            </tr>

        </table>

    </form>

    <p id="demo"></p>

    <script>

        var objArray = [];

        function addFunction(event) {

            event.preventDefault();

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

            var carPlateNumber = document.getElementById("carPlate").value;

            var hours = parseFloat(document.getElementById("hours").value);

            var obj = new Object();

            obj.CarPlate = carPlateNumber;

            obj.ParkingHours = hours;

            obj.Charges = calculateCharges(hours);

            objArray.push(obj);

            objArray.sort(function (a, b) { // ascending order sorting function based on ParkingHours

                return a.ParkingHours - b.ParkingHours; // return positive value if a.ParkingHours is greater than b.ParkingHours

            });

            objArray.reverse(); // to make elements in the array to be in descending based on the Parking Hours

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

                document.getElementById("demo").innerHTML += "Car Plate Number: " + objArray[i].CarPlate + "  Parking Hours: " + objArray[i].ParkingHours + " Charges: " + objArray[i].Charges + "<br>";

5            }

            document.getElementById("carPlate").value = "";

            document.getElementById("hours").value = "";

        }

        function calculateCharges(hours) {

            var charge = 0.0;

            if (hours > 0.0 && hours <= 3.0)

                charge = 2.0;

            else if (hours > 3.0) {

                charge = 2.0 + 0.5 \* Math.ceil(hours - 3.0);

                if (charge > 10.0)

                    charge = 10.0;

            }

            return charge;

        }

    </script>

</body>

</html>

What is an Array of Objects?

Unlike traditional array which store values like string, integer, Boolean, etc. array of objects stores objects. **The array elements store the location of reference variables of the object**.

Example:

var obj=[];

Object literal

obj[0]={ FullName:"John", ParkingHours:2,Charges:2.0};

obj[1]= { FullName:"Mary", ParkingHours:4,Charges:2.5};

Object constructor

var obj = new Object();

obj.FullName = "John”;

obj.ParkingHours = 2;

obj.Charges = 2.0;

obj = new Object();

obj.FullName = "Mary”;

obj.ParkingHours = 4;

obj.Charges = 2.5;

A diagram of a diagram

Description automatically generated

Basic Array Sorting

A screenshot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

A diagram of a graph

Description automatically generated

How to sort an array of objects by a property value in JavaScript

You want to order it by the parking hours, in **ascending** order. You can use the sort() method of Array, which takes a callback function, which takes as parameters 2 objects contained in the array (which we call a and b):

objArray.sort(function (b, a) {

return b.ParkingHours - a.ParkingHours;

});

When we return > 0, the function communicates to sort() that the object **a** takes precedence in sorting over the object **b**. Returning < 0 would do the opposite.