JAVASCRIPT:

1.Accessing a function without (), will return the function definition:

function toCelsius(f) { return (5/9) \* (f-32); }

2.Do Not Declare Strings, Numbers, and Booleans as Objects!

When a JavaScript variable is declared with the keyword "new", the variable is created as an object:

var x = new String(); // Declares x as a String object

var y = new Number(); // Declares y as a Number object

var z = new Boolean(); // Declares z as a Boolean object

3.objects

var person = {

firstName: "John",

lastName : "Doe",

id : 5566,

fullName : function() {

return this.firstName + " " + this.lastName;

}

};

4. In HTML, the global scope is the window object:

All global variables belong to the window object.

5. for global variables(declared inside a function), dnt use var keyword ( by default the type will bw windows.obj)

6.var x = new String("John");

var y = new String("John");

// (x == y) is false because objects cannot be compared.

JavaScript objects cannot be compared.

7.substr() Extracts a part of a string from a start position through a number of characters

substring() Extracts a part of a string between two specified positions

8.charAt() Returns the character at the specified index (position)

charCodeAt() Returns the Unicode of the character at the specified index

concat() Joins two or more strings, and returns a copy of the joined strings

fromCharCode() Converts Unicode values to characters

indexOf() Returns the position of the first found occurrence of a specified value in a string

lastIndexOf() Returns the position of the last found occurrence of a specified value in a string

localeCompare() Compares two strings in the current locale

match() Searches a string for a match against a regular expression, and returns the matches

replace() Searches a string for a value and returns a new string with the value replaced

search() Searches a string for a value and returns the position of the match

slice() Extracts a part of a string and returns a new string

split() Splits a string into an array of substrings

substr() Extracts a part of a string from a start position through a number of characters

substring() Extracts a part of a string between two specified positions

toLocaleLowerCase() Converts a string to lowercase letters, according to the host's locale

toLocaleUpperCase() Converts a string to uppercase letters, according to the host's locale

toLowerCase() Converts a string to lowercase letters

toString() Returns the value of a String object

toUpperCase() Converts a string to uppercase letters

trim() Removes whitespace from both ends of a string

valueOf() Returns the primitive value of a String object

9.The two methods, indexOf() and search(), are equal.

They accept the same arguments (parameters), and they return the same value.

The two methods are equal, but the search() method can take much more powerful search values.

10.slice(start, end)

substring(start, end)

substr(start, length)

The difference is that substring() cannot accept negative indexes.

11.Math.random(); // returns a random number

Math.min(0, 150, 30, 20, -8, -200); // returns -200

Math.max(0, 150, 30, 20, -8, -200); // returns 150

Math.round(4.7); // returns 5

Math.round(4.4); // returns 4

Math.ceil() rounds a number up to the nearest integer:

Math.ceil(4.4); // returns 5

Math.floor() rounds a number down to the nearest integer:

Math.floor(4.7); // returns 4

Math.floor() and Math.random() can be used together to return a random number between 0 and 10:

Math.floor(Math.random() \* 11); // returns a random number between 0 and 10

abs(x) Returns the absolute value of x

acos(x) Returns the arccosine of x, in radians

asin(x) Returns the arcsine of x, in radians

atan(x) Returns the arctangent of x as a numeric value between -PI/2 and PI/2 radians

atan2(y,x) Returns the arctangent of the quotient of its arguments

ceil(x) Returns x, rounded upwards to the nearest integer

cos(x) Returns the cosine of x (x is in radians)

exp(x) Returns the value of Ex

floor(x) Returns x, rounded downwards to the nearest integer

log(x) Returns the natural logarithm (base E) of x

max(x,y,z,...,n) Returns the number with the highest value

min(x,y,z,...,n) Returns the number with the lowest value

pow(x,y) Returns the value of x to the power of y

random() Returns a random number between 0 and 1

round(x) Rounds x to the nearest integer

sin(x) Returns the sine of x (x is in radians)

sqrt(x) Returns the square root of x

tan(x) Returns the tangent of an angle

12.Array adding a new element

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

fruits.push("Lemon"); // adds a new element (Lemon) to fruits

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

fruits[fruits.length] = "Lemon"; // adds a new element (Lemon) to fruits

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

fruits[10] = "Lemon"; // adds a new element (Lemon) to fruits

13.

Arrays with named indexes are called associative arrays (or hashes).

JavaScript does **not** support arrays with named indexes.

In JavaScript, **arrays** always use **numbered indexes**.

14. The valueOf() method returns an array as a comma separated string.

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

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

The **join()** method also joins all array elements into a string.

It behaves just like toString(), but you can specify the separator:

var fruits = ["Banana", "Orange","Apple", "Mango"];  
document.getElementById("demo").innerHTML = fruits.join(" \* ");

15. For JavaScript arrays, valueOf() and **toString()** are equal.

Shifting Elements

Shifting is equivalent to popping, working on the first element instead of the last.

The **shift()** method removes the first element of an array, and "shifts" all other elements one place down.

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.shift();            // Removes the first element "Banana" from fruits

The **unshift()** method adds a new element to an array (at the beginning), and "unshifts" older elements:

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.unshift("Lemon");    // Adds a new element "Lemon" to fruits

delete fruits[0];      // Changes the first element in fruits to **undefined**

Using **delete** on array elements leaves undefined holes in the array. Use pop() or splice() instead.