**Js interview topics**

1.JavaScript:

JavaScript is a programming language that executes on the browser. It turns static HTML web pages into interactive web pages by dynamically updating content, validating form data, controlling multimedia, animate images, and almost everything else on the web pages.

JavaScript is the third most important web technology after HTML and CSS. JavaScript can be used to create web and mobile applications, build web servers, create games, etc.

Syntax:

<script>

Write the JavaScript code here..

</script>

2.Variables are followed by the Keywords (var, let, const)

Syntax:

Variablekeyword variablename=value

For example:

Var name=”Nandhini” //string

Var num=123 //number

Var null=false //Boolean

Var username; //undefined

Var obj = (userID , abc,false,loc:)

**var :** This keyword is used to declare variable globally. If you used this keyword to declare variable then the variable can accessible globally and changeable also.

Eg:

Var name = “priya” (output=priya)

Name = “padma” (reassigning of variables //output=padma)

Var name = “shalu” (redeclaring the variable with the same name // output=shalu)

**Let:** Variables defined with let cannot be redeclared.

You cannot accidentally redeclare a variable. Variables defined with let must be Declared before use.

Variables defined with let have Block Scope.

Eg:

Let num=54 (output = 54)

Num = 45 ( reassigning of values to let variable // output=45)

Let num = 34(redeclaring of let variable // ouput=”identifier ‘num’ has already declared”)

**Const:** It is static. I t cant be redeclared or reassigned.

Eg:

Const num=123(output=123)

Num=456(reassigning of const variable //output = assignmrnt to constant variable)

Const num=789(rdeclaring of comst variable // output=’num’ has been already taken)

3.Callback function:

Callback function is declared inside one function and it will be executed according to the sequence order it called.

Eg – function firstName(){

secondName()

thirdName()

return "Myname"

}

function secondName(){

console.log("Nandhini")

}

function thirdName(){

console.log("hi")

}

console.log(firstName())

OUTPUT = Nandhini

hi

Myname

4.Promise:

Promises will return the value if it success by using resolve() and will return the error values using reject().

**Syntax –** var a = new Promise(function(resolve, reject)){

resolve()

reject()

}

a.then{

function.value(){

}

function.error(){

}

}

Eg - let a = new Promise((resolve, reject) => {

resolve("welcome")

})

let b = new Promise((resolve, reject) => {

resolve("thank you")

})

Promise.all([a,b]).then((val) => {

console.log(val)

})

OUTPUT = [ 'welcome', 'thankyou ' ]

**all() –** Will display output if all function are resolved

**allSettled() -** Will display output with status and value

**Eg – Output for above declared promise** [

{ status: 'fulfilled', value: 'welcome' },

{ status: 'fulfilled', value: 'thankyou' }

]

**any() –** will return the first success value either resolve or reject

**Eg - Output for above declared promise ->** welcome

**race() –** it will not check success or fail and return first executed promise function.

5.Hoisting:

Using of variable or function before it declared is hoisting.

**Eg –**console.log(a) // undefined

var a = 50; // 50 will be stored in a

console.log(a) //50

6.Ternary Operator

**? –** Symbol for ternary operator in JS.

It is used to declare conditions like if, is else

**Syntax –** condition ? true : false

**Eg -** function name1(){

return "Hi"

}

function name2(){

return "Welcome"

}

let auth = true;

console.log(auth ? name1() : name2());

OUTPUT = Hi

7.this keyword:

this keyword refers to different objects.

**Eg -** var name = {

firstName: "nandhini",

lastName: "rathiinam",

fullName : function() {

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

}

};

console.log(person.fullName())

OUTPUT =nandhini rathianam

8.String Methods:

String methods in JS is used to perform various operations with the string values declared.

**length** – used to find the length of string

**Eg - v**ar name = "hello"

console.log(name.length) //OUTPUT = 5

**split(“”) -** Splits the every letter of variable into seperate index of array

**Eg** - var splitExample = "nandhini rathianam"

console.log(splitExample.split("")) // OUTPUT [

// n,a,n,d,h,i,n,i,

//r,a,t,h,i,n,a,m

**split(“ “) - Sp**lits the every words of variable into seperate index of array

**Eg -**  var splitExample = "hello good morning"

console.log(splitExample.split(" ")) // OUTPUT [ 'hello', 'good', 'morning' ]

**slice()** – used to slice the part of content from variable

**Eg** - var splitExample = "The book of woman"

**cons**ole.log(splitExample.slice(4,7)) //OUTPUT = book

**includes()** - return true or false(will check also the Caps)

**Eg** - var name = "nandhini"

console.log(name.includes('i')) // OUTPUT = true

9.Array Methods:

* Array methods in JS is used to perform various operations with the array values declared.
* **Map() –** returns true if the given value is present in array and returns false for remaining index values.

**Eg -** var colours = ["red", "yellow", "blue", "pink"]

console.log(cars.map((d) => c=== 'blue')) OUTPUT = [ false, false, true, false ]

* **Slice()** - used to slice the part of content from array

**Eg -** var names = [nandhu, "shalu", "mano", "padma"]

var slicedArray = names.slice(2,4)

console.log(slicedArray)

OUTPUT = [ 'shalu', 'padma' ]

* **reverse()** - to reverse the values of array

**Eg** - var names =[nandhu, "shalu", "mano", "padma" ]

console.log(names.reverse())

OUTPUT = [ 'padma’,’mano’,’shalu’,’nandhu’]

* **contact() -** To join the existing array with extra given values

**Eg -** var names=[nandhu, "shalu", "mano", "padma" ]

var newnames = names.concat(["gokul", "siva"]);

console.log(newnames)

OUTPUT = [“nandhu”, "shalu", "mano", "padma" ,”gokul”,”siva”]

* **toString()** - Convert an array to a string

**Eg** - var names =[nandhu, "shalu", "mano", "padma" ]

console.log(namess.toString())

OUTPUT = nandhu,shalu,mano,padma

10.Synchronous JavaScript and Asynchronous JavaScript

**Synchronous JavaScript:** As the name suggests synchronous means to be in a sequence, i.e. every statement of the code gets executed one by one. So, basically a statement has to wait for the earlier statement to get executed.

**Example:**

|  |
| --- |
| <script>      document.write("Hi"); // First      document.write("<br>");    document.write("webshine") ;// Second      document.write("<br>");   document.write("How are you"); // Third |

</script>

**Output:**

**Hi**

**Webshine**

**How are you**

**Asynchronous JavaScript:** Asynchronous code allows the program to be executed immediately where the synchronous code will block further execution of the remaining code until it finishes the current one. This may not look like a big problem but when you see it in a bigger picture you realize that it may lead to delaying the User Interface.

Let us see the example how Asynchronous JavaScript runs.

|  |
| --- |
| <script>      document.write("Hi");      document.write("<br>");        setTimeout(() => {          document.write("Let us see what happens");      }, 2000); |

document.write("<br>");

    document.write("End");

    document.write("<br>");

</script>

**Output**

**Hi**

**End**

**Let us se what happens**