# What is Javascript?

# What is Typescript?

# What is difference between passing value and reference in Javacript?

Pass By Value: In Pass by value, function is called by directly passing the value of the variable as an argument. So any changes made inside the function does not affect the original value.

In Pass by value, parameters passed as an arguments create its own copy. So any changes made inside the function is made to the copied value not to the original value .

Pass by Reference: In Pass by Reference, Function is called by directly passing the reference/address of the variable as an argument. So changing the value inside the function also change the original value. In JavaScript array and Object follows pass by reference property.

In Pass by reference, parameters passed as an arguments does not create its own copy, it refers to the original value so changes made inside function affect the original value.

In Pass by Reference, we are mutating the original value. when we pass an object as an arguments and update that object’s reference in the function’s context, that won’t affect the object value. But if we mutate the object internally, It will affect the object .

# What functions are there in Javascript?

# What is immediately invoked function?

Immediately Invoked Function Expression, or IIFE for short is a function that is executed immediately after it’s created.

# What is prototype in Javascript? What is the meaning of this?

A function's this keyword behaves a little differently in JavaScript compared to other languages. It also has some differences between strict mode and non-strict mode.

In most cases, the value of this is determined by how a function is called (runtime binding). It can't be set by assignment during execution, and it may be different each time the function is called. ES5 introduced the bind() method to set the value of a function's this regardless of how it's called, and ES2015 introduced arrow functions which don't provide their own this binding (it retains the this value of the enclosing lexical context).

var obj = {

  name: 'Alan Harper',

  getName: function() {

    console.log(this.name)

  }

}

var getName = obj.getName;

var obj2 = {name: 'Charlie Harper', getName}

obj2.getName();

# What is the difference between call() and apply() in Javascript?

# What is the difference between Local Scope and Global Scope Variable In JavaScript?

Global scope — Global scope contains all of the things defined outside of all code blocks. A code block simply consists of grouped statements inside curly braces ({ }). if statement, loops, function are examples of structure that create a code block. A global variable has global scope. A global variable is accessible from anywhere in the code.

Local Scope — Local scope contains things defined inside code blocks. A local variable has local scope. A local variable is only accessible where it’s declared.

# What is the difference between var and let in JavaScript?

var and let are both used for variable declaration in javascript but the difference between them is that var is function scoped and let is block scoped.

It can be said that a variable declared with var is defined throughout the program as compared to let.

for(var i=0; i < 10; i++) {

  setTimeout(() => console.log(i), 0)

}

for(let i=0; i < 10; i++) {

  setTimeout(() => console.log(i), 0)

}

What is output?  
  
var: 9, 9, 9, …, 9  
let: 0, 1, 2, 3, …, 9

# What loop functions are there in Javascript?

* [for statement](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops_and_iteration#for_statement)
* [do...while statement](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops_and_iteration#do...while_statement)
* [while statement](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops_and_iteration#while_statement)
* [labeled statement](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops_and_iteration#labeled_statement)
* [break statement](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops_and_iteration#break_statement)
* [continue statement](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops_and_iteration#continue_statement)
* [for...in statement](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops_and_iteration#for...in_statement)
* [for...of statement](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops_and_iteration#for...of_statement)

loops using map, filter, reduce and find

# Write recursive function as following

console.log(multiply(2)(3)(4)); //24

# Write anagrams finder in Javascript

isAnagram("Mary", "Yram") //TRUE

# Write bigger function

returnsBigger(3, 4, 5); //5

const returnsBigger = (a,b,c) => Math.max(a,b,c)

# 

# 