JavaScript Tricky Interview Questions

1.

```
console.log(2 + '2');//22
console.log(2 - '2');// - willl convert string to number internally so value is 0
```

2. Remove duplicates from nums array

```
let nums = [1,2,2,3];
console.log(new Set(nums));// {1, 2, 3 }
//above one is not array if you want convert it to array use sread operator
console.log([...new Set(nums)]);//[ 1, 2, 3 ]
```

```
3. let func = function(){

    {
        let l = "let";
        var v = "var";
    }
    console.log(v);//var
    console.log(l);//ReferenceError: l is not defined
}
func()
```

```
4. let func = function(){

    {
        (function(){
            let l = "let";
            var v = "var";
        })();

    }
    console.log(v);// v is not defined
    console.log(l);//ReferenceError: l is not defined
}
func()
```

```
5. console.log(5<6<7);//true

console.log(7>6>5);//false
// in first statement, the reason is 5<6 is true so,
```

```
console.log(true<7);//in programming true is one so console.log(1<7); returns
true
//in second statement, the reason is 7>6 is true so,
console.log(true>7);//in programming true is one so console.log(1>7); returns
false
```

```
6. var a =function(){
    return arguments;
}
console.log(a("hi"));//{ '0': 'hi' }
```

```
7. var a =()=>
    arguments;
console.log(a("hi"));//{ '0': {} }
//in arrow function arguments not bind with function default
```

```
8. var a =(...n)=>
    n;
console.log(a("hi"));//{ hi }
```

```
9. let x = function(){
    return
    {
        message:"hi"
    }
}
console.log(x());//undefined.the above return and objet both treat as a seperate
statments like return ; {message:"hi"}, so
```

```
10. let x = function(){
    return{
        message:"hi"
    }
}
console.log(x());//{ message: 'hi' }
```

```
11. let profile={
    name:"chandra"
```

```
}
profile.age="26";
//here i want to restrict this object to add properties so, we an use
Object.freeze(profile);//When we use free we can edit/ add any property
profile.address="abc";//this will not add
console.log(profile);//{ name: 'chandra', age: '26' }
```

```
12. let profile={
    name:"chandra"
}
profile.age="26";
//here i want to restrict this object to add properties so, we an use
Object.seal(profile);//we can edit property but we can not add new
profile.name="abc";//this will update
profile.address="xyz";//this will not add
console.log(profile);//{ name: 'abc', age: '26' }
```

```
13. let profile={
    name:"chandra",
    age:3
}
//now the question is i should able modilfy name but not age how ?

let profile = {
    name:"chandra"
}
Object.defineProperty(profile,'age',{
    value:3,
    writable:false
});

profile.name = "xyz";
profile.age = 5;//this will not work

console.log(profile);//Objectname: "xyz"age: 3__proto__: Object
```

```
14. console.log(Math.max(1,2));//2
//the meaning of above statement is the first value 1 is compared with the lowest
value called negative infinity
//so with -infinity it will compare with 1 and then 2 like that it will work
```

```
console.log(Math.max());//-Infinity, there is no value provided so default value is-infinity is the answer
```

```
15. console.log(Math.min(1,2));//1

//the meaning of above statement is the first value 1 is compared with the maximum value called infinity
//so with infinity it will compare with 1 and then 2 like that it will work

console.log(Math.min());//Infinity, there is no value provided so default value is infinity is the answer
```

```
16. const x =[1,2,3];

x[-1]=-1;
console.log(x);//[1, 2, 3, -1: -1]
console.log(x[x.indexOf(10000)]);//-1
//in above indexOf will be used for search, if element found it return positive, if element not found it returns negative one. so
//here xonsole.log(x[-1]); returns -1
```

```
17. const arr = [1,2,15,30,5,45,7];

console.log(arr.sort());//[ 1, 15, 2, 30, 45, 5, 7 ] by default javascript treat as strings
console.log(arr.sort((a,b)=>{
    return a>b
}));//[ 1, 2, 5, 7, 15, 30, 45 ]
console.log(arr.sort((a,b)=>{
    return a<b
}));//[ 45, 30, 15, 7, 5, 2, 1 ]
```

```
18. //let i =?

console.log(i*i);//0
console.log(i+1);//1
console.log(i-1);//-1
console.log(i/i);//1
//want print output like above what is i value
//i value is Number.MIN_VALUE
console.log(i);//5e-324
```

```
19. let x = [1,2,3]+[4,5,6];
console.log(x);//"1,2,34,5,6"
```

```
20. let x = [...[1,2,3],...[4,5,6]];
console.log(x);//[ 1, 2, 3, 4, 5, 6 ]
```

```
21. let x = String([...[1,2,3],...[4,5,6]]);
console.log(x);//"1,2,3,4,5,6"
```

```
22. console.log(5555555555555555);//5555555555555600
console.log(Number.MAX_SAFE_INTEGER);//9007199254740991, if you exceed this
javascript automatically placed as zeros
```

```
23. (function(){
    let a= b =100;
})();
console.log(b);//100
console.log(a);//undefined
//IIFE makes as a block scope here
//in javascript the above statement meaning is a is defined as a let but b is
undefined so it treat as global so the b value exist outside and a value is not.
```

```
24. console.log(NaN === NaN);//fasle
//Because there are many ways to represent a NaN, it makes sense that one NaN
will not be equal to another NaN. Still, Nan is not a Nan
```

```
25. console.log([]+[]);
//here + sigh converts array to string  the out put will be empty string
```

```
26. console.log({}+[]);//[object Object]
```

```
27. <div contenteditable="true">hello</div>
//we can edit this content in web page
```

28.

```
> document.body.contentEditable=true
< true
>
```

If we do like this we can edit any website

```
29. function y(){
    console.log(this.length);
}
x={
    length:5,
    method:function y(){
        arguments[0]();//here we send y,1 arguments to atcual y function so the
leth is 2
    }
};

x.method(y,1);
```

```
29. const x = "constructor";

console.log(x[x](01));//1

//const x = new String("constructor");
//x[x] = x['constructor'] is eqaul to x.constructor
//generally string is class, so every calss having a constructor so it returns
String() function so

console.log(new String(01));// returns 1
```

```
30. console.log(0.1+0.2);//0.30000000000000004
```

```
31. console.log(("hi").__proto__);//String {"", length: 0, constructor: ƒ, anchor:
ƒ, big: ƒ, blink: ƒ, …}

//because hi is a string it prints proto is string now,
console.log(("hi").__proto__.__proto__);//{constructor: ƒ, __defineGetter__: ƒ,
  defineSetter__: ƒ, hasOwnProperty: ƒ, __lookupGetter__: ƒ, …}
```

```
//in above line top level proto is constructor prints
console.log(("hi").__proto__.__proto__.__proto__);//null, obove construtor
nothing will be there
```

```
31. //write a function to return total number of arguments
//dont use loops

let x = function(){
    return [].slice.call(arguments).length;
}

console.log(x(1,2,3,4,5))//5
```

32.