React Js Interview Question

- What is React js
- What is difference between virtual dom and shallow dom, dom in React js
- What is controlled and uncontrolled component in React js
- What is hooks in React js
- What is jsx, babel, webpack
- What is Redux
- What is reducer, action, store in Redux
- What is middleware in Redux
- Explain data flow in Redux
- What is Redux-Thunk
- What is Redux-Saga, Difference between Redux-thunk and Redux-saga
- Difference between class component and function component
- How can we implement componentWillUnmount in function component
- useEffect,UseState, useMemo.useCallback hooks in Details
- Explain lifecycle method in React js
- What is difference between export default and export in React js
- What is portal in React js

- What is reconciliation in React js
- What is useRef in React js
- What is server side render in React js
- What is useStrict in React js
- What is fragment in React js
- What is react router in React js
- What is node module in React js
- What is the default localhost server port in react js. how can we change the local server port
- What is high order component in React js
- What is pure component in React js
- What is difference state and props in React js
- How to optimize React js app
- What is difference between React js and Angular js
- What is prop drilling in React js how to overcome it
- What is context api in React js
- What is super, constructor, render function in React js

HTML interview question

- What is <!Doctype html> in Html5
- What is difference between div and span in Html
- What is semantic tags and non semantic tags in Html
- What is difference between html and html5
- What is Iframe tag in Html5
- What are the formatting tags in html
- What is difference <b> and <Strong> in html
- What is view port attribute in html
- What is attribute in html
- What is block level element and inline element in html
- What is difference between Html and Html5

Css Interview Question

- What is difference between css and css3
- What are the selector in css
- What is media query in css
- What is different position in css
- What is bom in css
- What is difference between PX,unit,em,rem in css
- What is flex box in css
- What is pseudo selector in css
- How to make website responsive
- What are breakpoint for viewport responsive device
- Why we use box-sizing in css

Javascript interview Question

- What is Ecmascript in Javascript
- What is difference between let ,cons and var
- What is spread operator, Rest operator , default parameter
- What is deep copy and shallow copy in Javascript
- What is promise , callback function , async await in in Javascript
- What is difference between promise and callback in Javascript
- What is event bubbling and event capturing in Javascript
- What is higher order function in Javascript
- Explain different-2 types of function in Javascript
- What is arrow function in Javascript
- Why we use call, apply bind method in Javascript
- How many way to create object in Javascript

- What is prototype inheritance in Javascript
- What is typescript
- What are the array method , string method
- What is difference between java and javascript
- What is throttling and debouncing in js
- What is Null and undefined in javascript
- What are the falsy values in javascript
- What is execution context, event loop ,stack, call queue,microtask queue in Javascript
- What is setTimeOut and setInterval in Javascript
- What is object.seal and object.freeze in Javascript
- What is difference between map and set in Javascript
- What is Weakmap and Weakset in Javascript
- What is sessionStorage, localStorage , cookie,
- Write a program to sort an array
- What is use of json.stringify and json parse() method in Javascript
- What are is map, filter , reducer in javascript
- What is generator function in Javascript
- How to stop event propagation in Javascript
- What is closure in Javascript
- What is housing in Javascript

- What is dead zone in Javascript
- What is function currying in Javascript
- What is mutation observer in Javascript
- What is memorization in javascript

- **Write a program to find element occurence in array**

```
const arr = [1,1,2,3,1,4]
const count = {};

for (const element of arr) {
 if (count[element]) {
   count[element] += 1;
 } else {
   count[element] = 1;
 }
}
```

console.log(count); // 👉 {1: 3, 2: 1, 3: 2}

- **Write a program to remove duplicate item from array**

```
const arr = [1,2,3,4,1,2];
const b=[];
for(let i=0;i<arr.length;i++){

  if(b.indexOf(arr[i]) == -1){
      b.push(arr[i])
  }
}
console.log("removed array value",b)
```

```
 const arr = [1,2,3,4,1,2];
const b=[];

        arr.filter((dup)=>{
        if(b.indexOf(arr[dup]) == -1){
                b.push(arr[dup])
  }
  })
console.log("removed array value",b)
```

- **What will be output of that code**

Const firstname = fun();
Let name = ‘vivek’
Function fun(){
Return `my is ${name} malviya`
}
console.log(firstname);

- **Write a program for following output**

console.log("output with normal function",mul(2)(4)(6))

```
function mul(a) {
 return function (b) {
   return function (c) {
     return a * b * c;
   };
 };
}
console.log("output with normal function", mul(2)(4)(6));
```

- **Write a program for following output using arrow function**

```
const call = (a) => {
 return (b) => {
   return (c) => {
     return a * b * c;
   };
 };
};
console.log("output with arrow function", mul(2)(4)(6));
```

- **Write a program return resolve if value is less than 5 using Promise**
-

```
function fun(a){
        let myPromise = new Promise((myResolve, myReject)=> {
  let x = 0;

// The producing code (this may take some time)


  if (a < 7) {
```

```
    myResolve(`number is given ${a}`);
  } else {
    myReject("Error");
  }
});

myPromise.then((result)=>{
console.log(result)
})
}
fun(5);
```

- **What will be output for this program ?**

```
for (let i = 0; i < 5; i++) {
 setTimeout(function () {
   console.log(i);
 }, i * 1000);
}
```

And

```
for (var i = 0; i < 5; i++) {
  setTimeout(function () {
    console.log(i);
  }, i * 1000);
}
```

- **Write a program to multiply two number without using multiply Sign in Javascript**

```
function multiplay(a, b) {
let  answer = a;
 for (let i = 0; i < b - 1; i++) {
   answer += a;
 }

 return answer;
}
console.log(multiplay(5, 3));
```

- **Write a program sorting in javascript**

```
const arr = [3,2,5,4,1,0]
  for (let i = 0; i < arr.length; i++) {
        for (let j = i+1; j < arr.length; j++) {
           if(arr[i] < arr[j]) {
             let  temp = arr[i];
               arr[i] = arr[j];
               arr[j] = temp;
           }
        }
    }
    console.log("Elements of array sorted in ascending order:");
     for (let i = 0; i < arr.length; i++) {
         console.log("Elements of array sorted in ascending order",
arr[i]);
    }
```

- **What will be output ?**

```
var num = 4;
function outer() {
 var num = 2;
 function inner() {
   num++;
   var num = 3;
```

```
   console.log("num", num);
 }
 inner();
}
outer();

function sayHi() {
 return (() => 0)();
}
```

- **What's the console output of?**

const a = {};
const b = { key: 'b' };
const c = { key: 'c' };
a[b] = 123;
a[c] = 456;
console.log(a[c]);

Answer : -
Object keys are automatically converted into strings.
We are trying to set an ***object as a key*** to object a, with the value of 123.
However, when we stringify an object, it becomes "[object Object]".
So what we are saying here, is that a["[object Object]"] = 123.
Then,

we can try to do the same again.
c is another object that we are implicitly stringifying.
So then, a["[object Object]"] = 456. Then, we log a[b],
which is actually a["[object Object]"].
We just set that to 456, so it returns 456. */

- **Write a program to make polyfill for map method in javascript**

```
Array.prototype.mymap = function (cb) {
 let temp = [];
 for (let i = 0; i < this.length; i++) {
   temp.push(cb(this[i]));
 }
 return temp;
};

     const arr = [2, 3, 4, 5];

const result = arr.mymap((num) => {
 return num * 5;
});
console.log("result", result);
```

- **Write a program to make polyfill for filter method in javascript**

```
Array.prototype.myFilter = function (cb) {
 let temp = [];
 for (let i = 0; i < this.length; i++) {
   if (cb(this[i], i, this)) temp.push(cb(this[i]));
 }
 return temp;
};

const arr = [2, 3, 4, 5];
console.log("arr", arr);

const data = arr.filter((num) => {
 return num > 2;
});
console.log("resultnumber", data);
```

- **Write a program to make polyfill for reduce method in javascript**

```
Array.prototype.myReducer = function (cb, initialValue) {
 var accumulator = initialValue;
 for (i = 0; i < this.length; i++) {
   accumulator = accumulator ? cb(this[i], i, this) : this[i];
 }
 return accumulator;
};

const arr = [2, 3, 4, 5];
const sumOfArray = arr.myReducer((accu, curr, index, arr) => {
 return (accu += curr);
}, 0);
console.log("polyfil", sumOfArray);
```