1. **Give an example where call apply bind is. required?**

The call() method is a predefined JavaScript method.

It can be used to invoke (call) a method with an owner object as an argument (parameter).

const person = {  
  **fullName**: function() {  
    return this.firstName + " " + this.lastName;  
  }  
}

const person1 = {  
  firstName:"John",  
  lastName: "Doe"  
}

person.fullName.call(**person1**);

The apply() method takes arguments as an **array**.

const person = {  
  fullName: function(city, country) {  
    return this.firstName + " " + this.lastName + "," + city + "," + country;  
  }  
}  
  
const person1 = {  
  firstName:"John",  
  lastName: "Doe"  
}  
  
person.fullName.apply(person1, ["Oslo", "Norway"]);

With the bind() method, an object can borrow a method from another object.

const person = {  
  firstName:"John",  
  lastName: "Doe",  
  display: function () {  
    let x = document.getElementById("demo");  
    x.innerHTML = this.firstName + " " + this.lastName;  
  }  
}  
  
let display = person.display.bind(person);  
setTimeout(display, 3000);

This example will display the person name after 3 seconds.

1. **What is the difference between readFile and readFileSync?**

The fs.readFileSync() method is an inbuilt application programming interface of fs module which is used to read the file and return its content.

In fs.readFile() method, we can read a file in a non-blocking asynchronous way, but in fs.readFileSync() method, we can read files in a synchronous way, i.e. we are telling node.js to block other parallel process and do the current file reading process. That is, when the fs.readFileSync() method is called the original node program stops executing, and node waits for the fs.readFileSync() function to get executed, after getting the result of the method the remaining node program is executed.

1. **What does process in node.js mean?**

The process object in Node.js is a global object that can be accessed inside any module without requiring it. There are very few global objects or properties provided in Node.js and process is one of them. It is an essential component in the Node.js ecosystem as it provides various information sets about the runtime of a program.

To explore we will use one of its properties which is called process.versions. This property tells us the information about Node.js version we have installed. It has to be used with -p flag.

Process also provides various properties to interact with. Some of them can be used in a Node application to provide a gateway to communicate between the Node application and any command line interface. This is very useful if you are building a command line application or utility using Node.js

* process.stdin: a readable stream
* process.stdout: a writable stream
* process.stderr: a wriatable stream to recognize errors

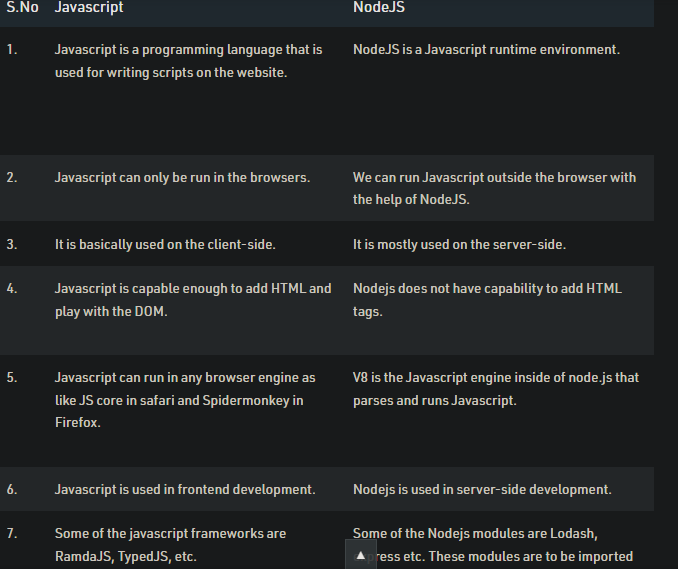
1. **Explain what node.js is?**

Node. js (Node) is an open source development platform for executing JavaScript code server-side. Node is useful for developing applications that require a persistent connection from the browser to the server and is often used for real-time applications such as chat, news feeds and web push notifications.

1. **What is the difference of JS from browser to JS on node.js?**

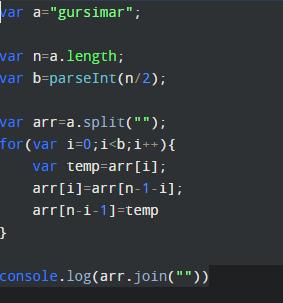
**1. [NodeJS](https://www.geeksforgeeks.org/introduction-to-nodejs/) :**   
NodeJS is a cross-platform and opensource Javascript runtime environment that allows the javascript to be run on the server-side. Nodejs allows Javascript code to run outside the browser. Nodejs comes with a lot of modules and mostly used in web development.

**2.**[**JavaScript**](https://www.geeksforgeeks.org/JavaScript-tutorial/)**:**   
Javascript is a [Scripting language](https://www.geeksforgeeks.org/introduction-to-scripting-languages/). It is mostly abbreviated as JS. It can be said that Javascript is the updated version of the ECMA script. Javascript is a high-level programming language that uses the concept of Oops but it is based on prototype inheritance.



1. **Write three different ways to reverse a string in Javascript? a. using inbuilt method, b. iteratively, c. recursively**
2. **Var arr=string.split(“”).reverse();**

**Console.log(arr.join(“”));**

1. 

**or**

function reverseString(str) {

var newString = "";

for (var i = str.length - 1; i >= 0; i--) {

newString += str[i];

}

return newString;

}

reverseString('hello');

1. function reverseString(str) {
2. if (str === "")
3. return "";
4. else
5. return reverseString(str.substr(1)) + str.charAt(0);
6. }
7. reverseString("hello");

**7. Write a program to check two objects are equal ( deep equal )**

**function deepEqual(object1, object2) {**

**const keys1 = Object.keys(object1);**

**const keys2 = Object.keys(object2);**

**if (keys1.length !== keys2.length) {**

**return false;**

**}**

**for (const key of keys1) {**

**const val1 = object1[key];**

**const val2 = object2[key];**

**const areObjects = isObject(val1) && isObject(val2);**

**if (**

**areObjects && !deepEqual(val1, val2) ||**

**!areObjects && val1 !== val2**

**) {**

**return false;**

**}**

**}**

**return true;**

**}**

**function isObject(object) {**

**return object != null && typeof object === 'object';**

**}**

const hero1 = {

name: 'Batman',

address: {

city: 'Gotham'

}

};

const hero2 = {

name: 'Batman',

address: {

city: 'Gotham'

}

};

deepEqual(hero1, hero2); // => true

* 1. **What is shallow equal?**

Shallow compare works by **checking if two values are equal in case of primitive types like string, numbers and in case of object it just checks the reference**. So if you shallow compare a deep nested object it will just check the reference not the values inside that object.