imp - we can acheive hoisting using var but can't achive using let/const.

Arrow functions are not hoisted as they are also function expressions where the function is assigned as a value to the variable.

1. What is the difference between `let`, `const`, and `var`?

Ans-1: let and const are introduced in ES6. earlier we only use var to declare variables. when we declare variable using var it become functional scope variable while let and const are block variable

let is used to declare that type of variable whom we want to modify later but we can't modify variables declared with const they are constant.

note- var is limited to function and let/const are limited to block.

2.What are the different data types in JavaScript?

Ans-2 number,boolean,string,null,undefined,arrays,object.

3. How do you check the type of a variable in JavaScript?

Ans-3 using typeof method.

ex - let a=10;

console.log(typeof(a));

4. What is hoisting in JavaScript?

Ans-4 in js we can use any variable before it's declaration.

Hoisting is a behavior in JavaScript where variable and function declarations are moved to the top of their containing scope during the compilation phase.

when js code runs first it declares the memory to all variable so it dosen't matter where the variable is declared. it will execute.

ex- b=10;

console.log(b);

var b;

Note - we can only acheive hoisting by using var initializer only.

5. What is the difference between `==` and `===`?

Ans-5: == check only the value whereas === checks for two things - value,data type

6. What is the use of the `this` keyword in JavaScript?

Ans-6 this keyword depends where we are calling it.it always refers an object.

if we are calling this in function it will return window object.

if we are calling this in object method we will get that object.

Note:- method of an object is when we create an function inside an object.

```
Ex- let detail = {
name:'Sagar',
        age:23,
        method: all(){
          return this;
           }
}
console.log(detail.method)
```

7. What is a closure? Can you provide an example?

Ans-7 when we have function inside an function so the inner function contains the lexical scope which means it can access variables of outer function.

Ex-

```
    function squre(){
            let n=4;
             function calc(){
               let res = n*n;
                       console.log(res);
             }
             calc();
            }
            console.log(squre());
```

8. Explain the difference between synchronous and asynchronous programming.

Ans-8 js is synchronous by nature but we can acheive asynchronous behaviour using promise and callback functions.

synchronous programming is when compiler reads the code from top to bottom while in asynchronous the block of code compiles after some time but it not affect the normal code it compiles side by side

9. What are callbacks? How do they work in JavaScript?

Ans-9 when we pass an function as an argument of another function this is called as callback.

Callbacks are commonly used to achieve asynchronous behavior, handle events, and perform tasks after the completion of an operation.

Ex- function squre(n){

return n*n;

}

function calc(a , fxn){

let c= a+10;

console.log(fxn(c));

}

calc(4, squre)

10.What is a Promise?

Ans-10 promises are the part of asynchronous programming in js.they did not return any output directly but they return after successfully compiling the code.

there are three phases of promise - pending,fullfill,reject

we use .then, .catch, .finally to get the output from an promise. and we declare Promise with new keyword and it takes two parameters one for fullfill and other for reject

```
Ex- new promise((fullfill,reject)=>{
        setTimeout(()=>{
        fullfill(
console.log('ok')
)
        reject(
console.log('Not OK')
)
        },2000)
})
                .then((res)=> console.log(res))
                .catch((err)=> console.log(err))
```

11. What is the event loop in JavaScript?

Ans-11 The event loop enables JavaScript to handle asynchronous operations.. js execute all operations in single thread.

there is call stack which keeps all operations in a queue and when a operation is completed it push another operation and pop the executed operation.

there is event queue which is responsible to send operations to call stack.

Note - Whenever there is async operation it sends it to corresponding api and sends back when the time ends.

12. Explain the concept of function hoisting.

Ans-12 Function hoisting is similer to variable hoisting in js we can call a function on top then after we can declare it.

when the code compiles the hoisting send the all initializes on top.We can hoist both function - function expression/normal function

Ex-

```
calling();

function calling(){
```

```
    console.log(called);
    }

calling1();

let calling1 = ()=>{
console.log('called');
};
```

13. How does prototypal inheritance work in JavaScript?

Ans-13. inheritance- one object trying to access the method and properties of another object

jb bhi hm js fxn/array/obj bnate hain toh js engine inbuild functions,methods unka sath attach kr deta hai jinko hm use kr skte hain.

Aur hm use .(dot) se access kr skte hain.

Note:- Object.getPrototypeOf() isse hm unhe dekh sakte hain.

Ex- let names = ['Sagar',23,31]

Object.getPrototypeOf(names)

14.What is the difference between `null` and `undefined`?

Ans-14 null is when we want to keep our variable empty so we assign it as null. Ex - let a = null

undefined error came when variable is defined but dosen't have any value. Ex- let a; console.log(a);

15. How do you create an object in JavaScript?

Ans-15 there are many ways to create objects in js-

first is object literals like normal object

```
    Ex- let name = {
    name:'Sagar',
        age:23
    }
```

second is we use object.create method in this we copy previous object and only change the details

Ex - let name={

name:'Sagar',age:23

}

let name1 = Object.create(name);

name1.name='Sagar1';

name1.age = 21

Third is using constructor function in this we make function with this keyword and use it with new keyword

Ex - function name(name,age){

this.name=name,

this.age = age

}

let name1 = new name('Sagar',23)

16. What is the purpose of the call/apply/bind method in JavaScript?

Ans-16 call- call allows an object to use the method of an another object.isme arguments bhi pass kr skte hain comma se saperate krke

apply- Apply method and call method are the same. The only difference is how we are passing parameters. We pass parameters in array form in apply.

bind - bind method does not directly execute the function. it makes a copy of that method and attached it with another obj. we can store it in any variable.

Ex- let name = {

name:'Sagar',

age:23,

detail: function(city,state){

```
return this.name +' ' + this.age + ' ' + city + ' ' + state;
}
}

let name2 ={
name:'k',
age:21
}

name.detail.call(name2, "Palwal", 'Haryana')
name.detail.call(name2,[ "Palwal", 'Haryana'])
let name3 = name.detail.call(name2, "Palwal", 'Haryana'); // It will return a function
console.log(name3());
```

17.Explain the concept of lexical scope.

Ans-17. jb ek function bnta hai toh voh apna ek environment bna leta hai jisme variables store hote hain agar uske ander ek aur fxn bnaye toh voh apna alag lexical environment bna leta hai.

Lexical Scoping:

Lexical scoping refers to the way variable names are resolved in nested functions. A function can access variables from its own scope, as well as from the scopes of its parent functions.

closure-

it is an ability of child function to access the lexical environment of parent function which means inner fxn can use variables of outer function.

18.What is the difference between `slice` and `splice` in JavaScript?

Ans-18 Slice and splice are js array methods slice is used to create a new array out of an old array.we can select start point and end point init.

Ex - let num = [1,2,3,4,5,6,7];

let num1 = num.slice(2,5);

console.log(num1);

in other hand splice is used to add values at certain index.it does not create new array it modifies old one.We can also delete Values

Ex- let num = [1,2,3,4,5,6,7];

num.splice(2,0,9,9,9) //first argument is index,second is howmany values we want to delete and other are values

console.log(num)

19.How do you iterate/loop over objects in JavaScript?

Ans-19 multiple methods of object iteration- for in method, object.keys , object.entries

method-1: by for in we can get keys and print them

```
let num2 = {id:2 , name:'Sagar' , active:true , age: 23};
        for(keys in num2){
           console.log(keys);
                    }
```

method-2:Object.keys can return only keys

```
 let num2 = {id:2 , name:'Sagar' , active:true , age: 23};
let num3 = Object.keys(num2);

num3.map((data)=>{console.log(data)});
```

method-3:Object.entries return an array of both keys and values which we can print.

```
let num2 = {id:2 , name:'Sagar' , active:true , age: 23};
let num3 = Object.entries(num2);

num3.map((data)=>{console.log(data[0] ,data[1])});
```

20.Explain the concept of a callback hell and how to avoid it.

Ans-20 if we have multiple callback functions and all are depanding on each other so we have to call them inside of each other which can be very hard to

debug and not easy to maintain code and it also decrease the redablity of the code.this is called callback hell.

To avoid it we use promises.

==========

---

2. ES6+ Features:

- Arrow functions => normal fxn without function declaration = ()=>{}
- Template literals => Used to concaadinate two strings
- Destructuring assignment
- Spread/rest operators
- Let and const
- Classes and inheritance
- Modules

Destructuring assignment => Allows use to extract data from array, obj and assign them into local variables

Ex- let arr = ['ok' , 23 , 'Sagar']

let[status , age,name]= arr;

Spread/rest operators =

spread is used to unpack/expand elements from an array or object and place them into another array or object

Ex-

```
let a =[1,3,5,44,5,8,5,0];
let b=[2,3,5];

let c=[...a,...b];
console.log(c);
```

rest is used in function argument which means we can pass as much as argument we want

Ex- function sum(...a){}

Classes => class is a blueprint/template to create an object.we create class with class keyword.

We use constructor function in this.We create newclass with new keyword.

```
Ex - class person{
constructor(Name,Age){
this.name = Name,
this.age = Age
}
}
let data = new person('Sagar',23)
```

3. Async JavaScript:

- Promises => pending,fullfill,rejected
- `async` and `await`
- Fetch API or XMLHttpRequest

`async` and `await` => We use async keyword to create asynchronous function in this we don't have to use .then instead of that

we use await keyword to wait for the output

Ex-function newdata(){

```
new Promise((resolve,reject)=>{
   setTimeout(()=>{
   let a=10;
   if(a>15){
      resolve(console.log('Fine'))
   }
   else{
      reject(console.log('Not Fine'))
   }
      },2000)
})
}

   async function data(){
      try {
   const data = await newdata();
}
 catch (error) {
   console.log(error)
}
}

data();
```

4. **DOM Manipulation:**

- Selecting and manipulating DOM elements -getElementById,className,Tagname
- Event handling- onClick, onLoad, onKeyUp , onKeyDown , onmouseHover , onKeyPress
- Event delegation
- DOM traversal

Event delegation - Event Delegation is basically a pattern to handle events efficiently. Instead of adding an event listener to each and every similar element,

we can add an event listener to a parent element and call an event on a particular target using the .target property of the event object.

Like bubbling

DOM traversal - DOM traversal is the process of navigating through the Document Object Model (DOM), which represents the structure of an HTML or XML document

methods - parentNode , ParentElement,children , childNodes , firstChild , lastChild, firstElementChild , LastElementChild , nextsibling, nextElementSibling,previousSibling

Ex - let a = document.getElementById('name').parentNode ;

```
console.log(a);
```

5. **Functional Programming:**

- First-class functions - When we can assign any function body to any variable or function call krte vakt pura function as an argument pass kr skte hain.js allow this.

- Higher-order functions -

- Map, filter, and reduce

- Immutability

Higher-order functions - jb function dusra function as a parameter le and return bhi function hi kare.

- Map, filter, and reduce = teeno hi array methods hain

filter=> used to filter/remove particular elements from an array and creates an new array.

Ex - let a = [1,2,3,4,5];

```
let b = a.filter((data)=>{
   if(data !=2){
      return data;
   }
});

console.log(b);
```

map => It applies a given function on all the elements of the array and returns the updated array. It is the simpler and shorter code instead of a loop.

reduce =>The reduce method is used to reduce an array to a single value.

Ex - var myArray = [1, 2, 3, 4, 5];

```
var sum = myArray.reduce(function(acc, cur) {
   return acc + cur;
```

```
}, 0);
console.log(sum); // 15
```

Immutability - if we can't change/modify the value directly we can change the copy of that. Object are mutable and strings are immutable.

Agar mutable objects ko immutable bnana ho toh hm in methods ka use kr skte hain -

1. Object.seal(name) => we can't add/delete it makes it immutable but we can change the value.

2. Object.freeze(name) => no add/delete/change

Ex- let ak = {name:'Sagar',age:23};

```
Object.freeze(ak);
ak.name = 'Kaushik';
console.log(ak);
```

6. Closures and Scopes:

- Understand how closures work => child part can access the lexical scope of parent function

- Lexical scoping => when an function creates it create a scope in which all the variables are stores this is called lexicals scope

7. Prototype and Inheritance:

- Prototypal inheritance

- `Object.create` and `Object.setPrototypeOf`

Object.create is a method in which old object copies and we can change only key values.

Object.setPrototypeOf ==>`Object.setPrototypeOf` is used to set the prototype of one object to be another object.

- This means that the properties and methods of the second object become accessible to the first object through the prototype chain.

Ex -
```
const animal = {
  makeSound: function() {
```

```
    console.log('Some generic sound');
  }
};

const dog = {
  breed: 'Labrador'
};

Object.setPrototypeOf(dog, animal);

console.log(dog);
```

8. Error Handling:

- `try`, `catch`, `finally` => try mein jisme error pakadna hai aur catch mein aur finally dono case mein chalega

- Custom errors => we can create custom error using throw new Error in js

Ex-
```
function sum(a,b) {
      if(a+b<3){
       throw new Error('Not valid')
       }
      }

 try{
    sum(1,1);
    console.log(sum);
    }
   catch(error){
   console.log(error);
    }
```

9. **Asynchronous Programming:**

- Callbacks - passing function as an argument

- Promises - used in async programming.pending,resolved,reject

- `async`/`await` - async se asynchronous fxn bnta hai aur await use krke call kr skte hain // callback hell se bhi bachata hai.

10. Event Loop: Responsible for execution of async program // call stack & Event queue

15. AJAX and Fetch API:

In simple terms, both AJAX (Asynchronous JavaScript and XML) and the Fetch API are used to make requests to a server and retrieve data without reloading the entire web page. However,

there are some key differences between them.

AJAX (Asynchronous JavaScript and XML):

1. Older Technology:

2. Complex API:

AJAX uses a more complex API, involving multiple objects and methods such as `XMLHttpRequest`. It can be a bit more verbose and harder to work with compared to newer alternatives.

3.Callbacks:

- AJAX relies heavily on callback functions to handle asynchronous operations. This can sometimes lead to callback hell, a situation where nested callbacks become hard to manage.

Fetch API:

1. Modern Approach:

- The Fetch API is a more modern and simpler way to make HTTP requests. It is designed to be more straightforward and easier to use than AJAX.

2. Promise-Based:

- Fetch is promise-based, meaning it uses JavaScript promises to handle asynchronous operations. This makes it easier to work with asynchronous code, especially with the introduction of `async/await` syntax.

3. Simpler Syntax:

- The Fetch API has a simpler and cleaner syntax compared to AJAX. It involves fewer objects and methods, making it more readable and easier to understand.

19. Browser Storage:

- LocalStorage, SessionStorage, Cookies

LocalStorage:

LocalStorage is designed for storing data persistently on a user's device. It remains available even after the user closes the browser.The storage limit is usually around 5-10 MB.

The data stays until explicitly cleared by the user or through script.

Ex- localStorage.setItem('Name','Sagar');

SessionStorage:

SessionStorage is similar to LocalStorage but is designed for temporary storage. It stays for the duration of the page session and is cleared when the user closes the browser or tab.

Ex- SessionStorage.setItem('Name','Sagar');

Cookies:

We Use Cookies when We need to exchange data between the client and server.Cookies are small pieces of data sent from a website and stored on the user's device. They are often used for tracking and authentication purposes.

Cookies have a smaller storage limit (usually 4 KB) compared to LocalStorage and SessionStorage.

Cookies have an expiration date, and some may be set to last only for the duration of the session (session cookies), while others persist for a specified period.

Ex- document.cookie = 'key=value; expires=Wed, 21 Dec 2022 12:00:00 UTC; path=/';

20. Other Concepts:

- Hoisting - variables declares on top on time of code compiling

- Strict mode

- Event Bubbling and Capturing - bubbling mein pehle inner child pe event lega then parent and capturing mein uska ulta

Strict mode => Strict mode in JavaScript is a set of rules that helps catch common coding errors and prevents the use of certain "bad" practices.

It's like a set of stricter rules that you can voluntarily opt into when writing your JavaScript code.

==================================OTHER===

///

BASIC

//////////////

. What are the differences between Java and JavaScript?

JavaScript is a client-side scripting language and Java is object Oriented Programming language, both of them are totally different from each other.

JavaScript: It is a light-weighted programming language (“scripting language”) used to develop interactive web pages.

It can insert dynamic text into the HTML elements. JavaScript is also known as the browser’s language.

Java: Java is one of the most popular and widely used programming languages.

It is an object-oriented programming language and has a virtual machine platform that allows you to create compiled programs that run on nearly every platform.

Java promised, “Write Once, Run Anywhere”.

2. What are JavaScript Data Types?

There are three major Data types in JavaScript.

Primitive

Numbers

Strings

Boolean

Symbol

Trivial

Undefined

Null

Composite

Objects

Functions

Arrays

3. Which symbol is used for comments in JavaScript?

Comments are used to prevent the execution of statements. Comments are ignored while the compiler executes the code. There are two types of symbols used to represent comments in JavaScript:

Double slash: It is known as a single-line comment.

// Single line comment

Slash with Asterisk: It is known as a multi-line comment.

/*

Multi-line comments

...

*/

4. What would be the result of 3+2+”7"?

Here, 3 and 2 behave like an integer, and “7” behaves like a string. So 3 plus 2 will be 5. Then the output will be 5+”7" = 57.

5. What is the use of the isNaN function?

The number isNan function in JavaScript is used to determine whether the passed value is NaN (Not a number) and is of the type “Number”. In JavaScript, the value NaN is considered a type of number. It returns true if the argument is not a number, else it returns false.

6. Which is faster in JavaScript and ASP script?

JavaScript is faster compared to ASP Script because JavaScript is a client-side scripting language and does not depend on the server to execute it but the ASP script is a server-side scripting language always dependable on the server.

7. What is negative infinity?

The negative infinity in JavaScript is a constant value that is used to represent the lowest available value. It means that no other number is lesser than this value. It can be generated using a self-made function or by an arithmetic operation. JavaScript shows the NEGATIVE_INFINITY value as -Infinity.

8. Is it possible to break JavaScript Code into several lines?

Yes, it is possible to break the JavaScript code into several lines in a string statement. It can be broken by using the backslash ‘\’.

For example:

```
document.write("A Online Computer Science Portal\ for Geeks")
```

The code-breaking line is avoided by JavaScript which is not preferable.

```
let gfg= 10, GFG = 5,

Geeks =

gfg + GFG;
```

9. Which company developed JavaScript?

Netscape developed JavaScript and was created by Brenden Eich in the year of 1995.

10. What are undeclared and undefined variables?

Undefined: It occurs when a variable has been declared but has not been assigned any value. Undefined is not a keyword.

Undeclared: It occurs when we try to access any variable which is not initialized or declared earlier using the var or const keyword. If we use ‘typeof’ operator to get the value of an undeclared variable, we will face the runtime error with the return value as “undefined”. The scope of the undeclared variables is always global.

11. Write a JavaScript code for adding new elements dynamically.

html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Document</title>
</head>

<body>
  <button onclick="create()">
    Click Here!
  </button>

  <script>
    function create() {
      let geeks = document.createElement('geeks');
      geeks.textContent = "Geeksforgeeks";
      geeks.setAttribute('class', 'note');
      document.body.appendChild(geeks);
    }
  </script>
</body>
</html>
```

12. What are global variables? How are these variables declared, and what are the problems associated with them?

In contrast, global variables are the variables that are defined outside of functions. These variables have a global scope, so they can be used by any function without passing them to the function as parameters.

Example:

```
javascript
let petName = "Rocky"; //Global Variable
myFunction();

function myFunction() {
   document.getElementById("geeks").innerHTML
      = typeof petName + "- " +
      "My pet name is " + petName;
}

document.getElementById("Geeks")
   .innerHTML = typeof petName + "- " +
   "My pet name is " + petName;
```

It is difficult to debug and test the code that relies on global variables.

13. What do you mean by NULL in JavaScript?

The NULL value represents the no value or no object. It can be called an empty value/object.

14. How to delete property-specific values?

The delete keyword is used to delete the whole property and all the values at once like

```
let gfg={Course: "DSA", Duration:30};
delete gfg.Course;
```

15. What is a prompt box?

It is used to display a dialog box with an optional message prompting the user to input some text. It is often used if the user wants to input a value before entering a page. It returns a string containing the text entered by the user, or null.

16. What is the ‘this’ keyword in JavaScript?

Functions in JavaScript are essential objects. Like objects, they can be assigned to variables, passed to other functions, and returned from functions. And much like objects, they have their own properties. ‘this’ stores the current execution context of the JavaScript program. Thus, when it is used inside a function, the value of ‘this’ will change depending on how the function is defined, how it is invoked, and the default execution context.

17. Explain the working of timers in JavaScript? Also elucidate the drawbacks of using the timer, if any.

The timer is used to execute some specific code at a specific time or any small amount of code in repetition to do that you need to use the functions setTimout, setInterval, and clearInterval. If the JavaScript code set the timer to 2 minutes and when the times are up then the page displays an alert message “times up”. The setTimeout() method calls a function or evaluates an expression after a specified number of milliseconds.

18. What is the difference between ViewState and SessionState?

ViewState: It is specific to a single page in a session.

SessionState: It is user specific that can access all the data on the web pages.

19. How can you submit a form using JavaScript?

You can use document.form[0].submit() method to submit the form in JavaScript.

20. Does JavaScript support automatic type conversion?

Yes, JavaScript supports automatic type conversion.

Related Article: Commonly asked JavaScript Interview Questions | Set 1

//
//

Intermediate

////////////////////

1. What are all the looping structures in JavaScript ?

while loop: A while loop is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition. The while loop can be thought of as a repeating if statement.

for loop: A for loop provides a concise way of writing the loop structure. Unlike a while loop, for statement consumes the initialization, condition and increment/decrement in one line thereby providing a shorter, easy to debug structure of looping.

do while: A do-while loop is similar to while loop with the only difference that it checks the condition after executing the statements, and therefore is an example of Exit Control Loop.

2. How can the style/class of an element be changed?

To change the style/class of an element there are two possible ways. We use document.getElementByID method

```
document.getElementById("myText").style.fontSize = "16px;

document.getElementById("myText").className = "class";
```

3. Explain how to read and write a file using JavaScript?

The readFile() functions is used for reading operation.

readFile(Path, Options, Callback)

The writeFile() functions is used for writing operation.

writeFile(Path, Data, Callback)

4. What is called Variable typing in JavaScript ?

The variable typing is the type of variable used to store a number and using that same variable to assign a “string”.

```
Geeks = 42;

Geeks = "GeeksforGeeks";
```

5. How to convert the string of any base to integer in JavaScript?

In JavaScript, parseInt() function is used to convert the string to an integer. This function returns an integer of base which is specified in second argument of parseInt() function. The parseInt() function returns Nan (not a number) when the string doesn’t contain number.

6. Explain how to detect the operating system on the client machine?

To detect the operating system on the client machine, one can simply use navigator.appVersion or navigator.userAgent property. The Navigator appVersion property is a read-only property and it returns the string that represents the version information of the browser.

7. What are the types of Pop up boxes available in JavaScript?

There are three types of pop boxes available in JavaScript.

Alert

Confirm

Prompt

8. What is the difference between an alert box and a confirmation box?

An alert box will display only one button which is the OK button. It is used to inform the user about the agreement has to agree. But a Confirmation box displays two buttons OK and cancel, where the user can decide to agree or not.

9. What is the disadvantage of using innerHTML in JavaScript?

There are lots of disadvantages of using the innerHTML in JavaScript as the content will replace everywhere. If you use += like “innerHTML = innerHTML + ‘html'” still the old content is replaced by HTML. It preserves event handlers attached to any DOM elements.

10. What is the use of void(0) ?

The void(0) is used to call another method without refreshing the page during the calling time parameter “zero” will be passed.

11. What are JavaScript Cookies ?

Cookies are small files that are stored on a user’s computer. They are used to hold a modest amount of data specific to a particular client and website and can be accessed either by the web server or by the client’s computer. When cookies were invented, they were basically little documents containing information about you and your preferences. For instance, when you select the language in which you want to view your website, the website would save the information in a document called a cookie on your computer, and the next time when you visit the website, it would be able to read a cookie saved earlier.

12. How to create a cookie using JavaScript?

To create a cookie by using JavaScript you just need to assign a string value to the document.cookie object.

```
document.cookie = "key1 = value1; key2 = value2; expires = date";
```

13. How to read a cookie using JavaScript?

The value of the document.cookie is used to create a cookie. Whenever you want to access the cookie you can use the string. The document.cookie string keep a list of name = value pairs separated by semicolons, where name is the name of a cookie and the value is its string value.

14. How to delete a cookie using JavaScript?

Deleting a cookie is much easier than creating or reading a cookie, you just need to set the expires = “past time” and make sure one thing defines the right cookie path unless few will not allow you to delete the cookie.

15. What are escape characters and escape() function?

Escape character: This character is required when you want to work with some special characters like single and double quotes, apostrophes, and ampersands. All the special character plays an important role in JavaScript, to ignore that or to print that special character, you can use the escape character backslash “\”. It will normally ignore and behave like a normal character.

// Need escape character

document.write("GeeksforGeeks: A Computer Science Portal "for Geeks" ")

document.write("GeeksforGeeks: A Computer Science Portal \"for Geeks\" ")

escape() function: The escape() function takes a string as a parameter and encodes it so that it can be transmitted to any computer in any network which supports ASCII characters.

16. Whether JavaScript has a concept-level scope?

JavaScript is not concept-level scope, the variables declared inside any function have scope inside the function.

17. How generic objects can be created in JavaScript?

To create a generic object in JavaScript use:

var I = new object();

18. Which keywords are used to handle exceptions?

When executing JavaScript code, errors will almost definitely occur. These errors can occur due to the fault from the programmer’s side due to the wrong input or even if there is a problem with the logic of the program. But all errors can be solved by using the below commands.

The try statement lets you test a block of code to check for errors.

The catch statement lets you handle the error if any are present.

The throw statement lets you make your own errors.

19. What is the use of the blur function?

It is used to remove focus from the selected element. This method starts the blur event or it can be attached to a function to run when a blur event occurs.

20. What is the unshift method in JavaScript?

It is used to insert elements in the front of an array. It is like a push method that inserts elements at the beginning of the array.

//
//

Advance

//////////////

1. What is the ‘Strict’ mode in JavaScript and how can it be enabled?

Strict Mode is a new feature in ECMAScript 5 that allows you to place a program or a function in a “strict” operating context. This strict context prevents certain actions from being taken and throws more exceptions. The statement “use strict” instructs the browser to use the Strict mode, which is a reduced and safer feature set of JavaScript.

2. How to get the status of a CheckBox?

The DOM Input Checkbox Property is used to set or return the checked status of a checkbox field. This property is used to reflect the HTML Checked attribute.

```
document.getElementById("GFG").checked;
```

If the CheckBox is checked then it returns True.

3. How to explain closures in JavaScript and when to use it?

The closure is created when a child functions to keep the environment of the parent’s scope even after the parent’s function has already executed. The Closure is a locally declared variable related to a function. The closure will provide better control over the code when using them.

```
// Explanation of closure
function foo() {
   let b = 1;
   function inner() {
      return b;
   }
   return inner;
}
let get_func_inner = foo();


console.log(get_func_inner());
console.log(get_func_inner());
console.log(get_func_inner());
```

4. What is the difference between call() and apply() methods ?

Both methods are used in a different situation

call() Method: It calls the method, taking the owner object as argument. The keyword this refers to the 'owner' of the function or the object it belongs to. We can call a method that can be used on different objects.

apply() Method: The apply() method is used to write methods, which can be used on different objects. It is different from the function call() because it takes arguments as an array.

5. How to target a particular frame from a hyperlink in JavaScript ?

This can be done by using the target attribute in the hyperlink. Like

```
<a href="/geeksforgeeks.htm" target="newframe">New Page</a>
```

6. Write the errors shown in JavaScript?

There are three different types of errors in JavaScript.

Syntax error: A syntax error is an error in the syntax of a sequence of characters or tokens that are intended to be written in a particular programming language.

Logical error: It is the most difficult error to be traced as it is the error on the logical part of the coding or logical error is a bug in a program that causes to operate incorrectly and terminate abnormally.

Runtime Error: A runtime error is an error that occurs during the running of the program, also known as an exception.

7. What is the difference between JavaScript and Jscript?

JavaScript

It is a scripting language developed by Netscape.

It is used to design client and server-side applications.

It is completely independent of Java language.

Jscript

It is a scripting language developed by Microsoft.

It is used to design active online content for the word wide Web.

8. What does var myArray = [[]]; statement declares?

In JavaScript, this statement is used to declare a two-dimensional array.

9. How many ways an HTML element can be accessed in JavaScript code?

There are four possible ways to access HTML elements in JavaScript which are:

getElementById() Method: It is used to get the element by its id name.

getElementsByClass() Method: It is used to get all the elements that have the given classname.

getElementsByTagName() Method: It is used to get all the elements that have the given tag name.

querySelector() Method: This function takes CSS style selector and returns the first selected element.

10. What is the difference between innerHTML & innerText?

The innerText property sets or returns the text content as plain text of the specified node, and all its descendants whereas the innerHTML property sets or returns the plain text or HTML contents in the elements. Unlike innerText, inner HTML lets you work with HTML rich text and doesn’t automatically encode and decode text.

11. What is an event bubbling in JavaScript?

Consider a situation an element is present inside another element and both of them handle an event. When an event occurs in bubbling, the innermost element handles the event first, then the outer, and so on.

12. What will be the output of the following code?

```
let X = { geeks: 1 };

let Output = (function () {

   delete X.geeks;

   return X.geeks;

})();


console.log(output);
```

Here the delete will delete the property of the object. X is the object with the geek's property and it is a self-invoking function that will delete the geek's property from object X so the result will be undefined.

13. How are JavaScript and ECMA Script related?

JavaScript is the main language that has to maintain some rules and regulations which is ECMA Script, these rules also bring new features for the language JavaScript.

14. How to hide JavaScript code from old browsers that don't support JavaScript?

To hide the JavaScript codes from the old browsers that don't support JavaScript you can use

```
<!-- before <script> tag and another //--> after </script> tag
```

all the old browsers that will take that as a long comment of HTML. New browsers that support JavaScript will take that as an online comment.

15. What will be the output of the following code?

```
let output = (function(x) {

   delete x;

   return x;
```

```
})(0);

document.write(output);
```

The output will be 0. The delete operator is used to delete the operator of the object but the X is not the object here it is a local variable. The delete operator doesn't affect local variables.

16. In JavaScript, answer if the following expressions result in true or false.

```
"0" == 0   // true or false ?

"" == 0   // true or false ?

"" == "0"   // true or false ?
```

The result will be True for 1st and 2nd case and False for the 3rd case.

17. How to use any browser for debugging?

By pressing the F12 we can trigger the debugging mode of any browser and can view the result by taping the console.

18. What is javascript Hoisting?

When any interpreter runs the code then all the variables are re-hoisted to the top of the original scope. This method is applicable for declaration not for the initialization of a variable. This is known as a javascript Hoisting.

19. What is the syntax of 'Self Invoking Function' ?

The syntax for Self-Invoking Function: The last bracket contains the function expression.

```
(function () {
  return // body of the function
}());
```

20. How to use external JavaScript file in another JavaScript file?

You can use the below code to use external JavaScript code in another JavaScript file.

```
let script = document.createElement('script');

script.src = "external javascript file";
```

```
document.head.appendChild(script);
```