Java Script Interview Questions [Session: 2024-25]

1. What do you understand about JavaScript?

JavaScript is a popular web scripting language used for client-side and server-side development. Its code can be inserted into HTML pages and understood and executed by web browsers. JavaScript also supports object-oriented programming abilities.

2. What Are the Different Data Types in JavaScript?

JavaScript has six primitive data types:

- Number
- String
- Boolean
- Null
- Undefined
- Symbol

It also has two compound data types:

- Object
- Array

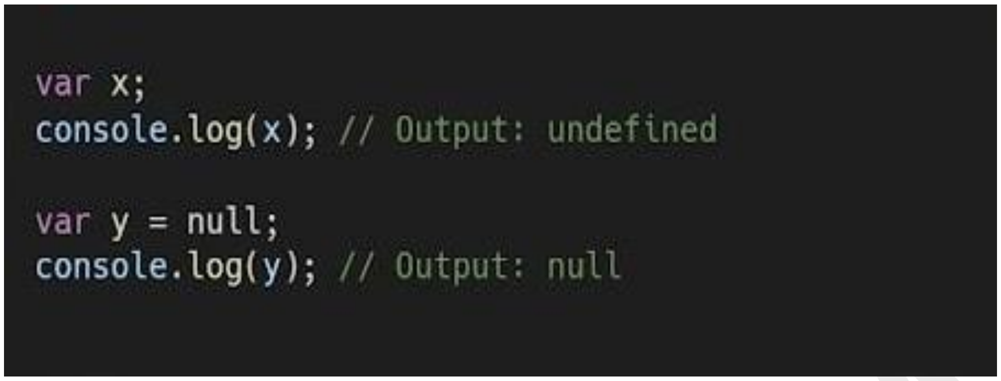
3. What Is Hoisting in JavaScript?

Hoisting is a JavaScript concept that refers to the process of moving declarations to the top of their scope. This means that variables and functions can be used before they are declared, as long as they are declared before they are used in a function.

4. What Is the Difference Between null and undefined?

null is an assignment value that represents no value or an empty value,
while undefined is a variable that has been declared but not assigned a value.

CYBROM



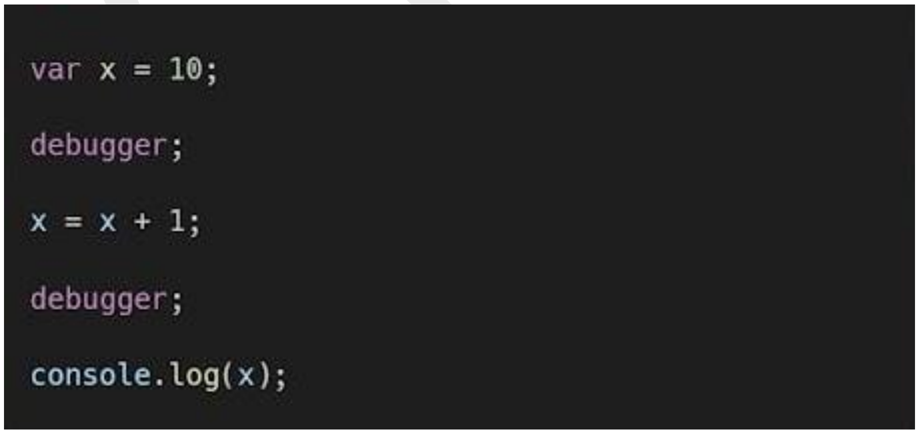
```
var x;
console.log(x); // Output: undefined

var y = null;
console.log(y); // Output: null
```

5. Why Do We Use the Word “debugger” in JavaScript?

The word “debugger” is used in JavaScript to refer to a tool that can be used to step through JavaScript code line by line. This can be helpful for debugging JavaScript code, which is the process of finding and fixing errors in JavaScript code. To use the debugger, you need to open the JavaScript console in your browser. Then, you can use debugger commands to comb through your code line by line.

It's essential to know debugging techniques as well as the more general ideas behind code optimization and speed improvement. In addition to operating smoothly, efficient code significantly enhances the user experience.

For example, the following code will print the value of the x variable at each step of the debugger.

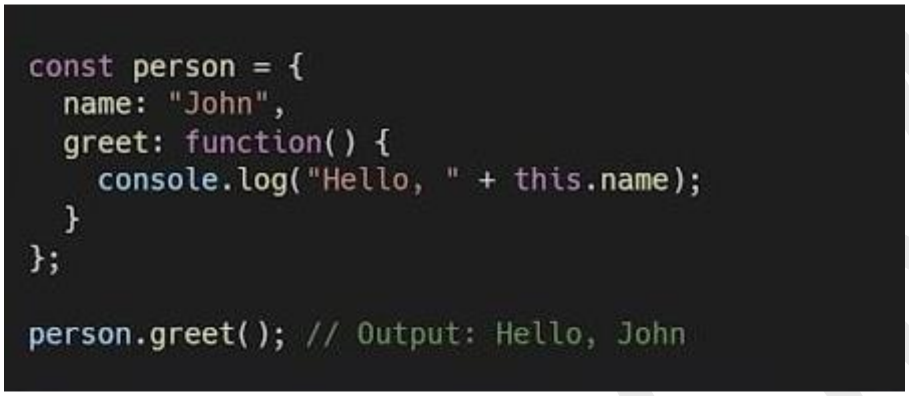


```
var x = 10;

debugger;

x = x + 1;

debugger;

console.log(x);
```

CYBROM

6. What Is the Purpose of the “this” Keyword in JavaScript?

The this keyword refers to the object that is executing the current function or method. It allows access to object properties and methods within the context of that object.

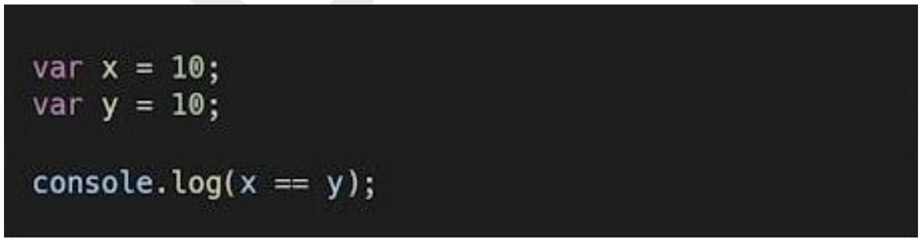


```
const person = {
  name: "John",
  greet: function() {
    console.log("Hello, " + this.name);
  }
};

person.greet(); // Output: Hello, John
```

7. What Is the Difference Between == and === Operators in JavaScript?

The equality == operator is a comparison operator that compares two values and returns true if they are equal. The strict equality === operator is also a comparison operator, but it compares two values and returns true only if they are equal and of the same type.

For example, the following code will return true, because the values of the x and y variables are equal.



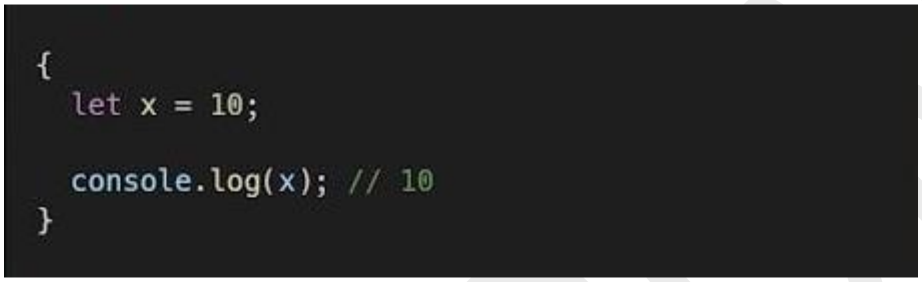
```
var x = 10;
var y = 10;

console.log(x == y);
```

CYBROM

8. What Is the Difference Between “var” and “let” Keywords in JavaScript?

The var and let keywords are both used to declare variables in JavaScript. However, there are some key differences between the two keywords.

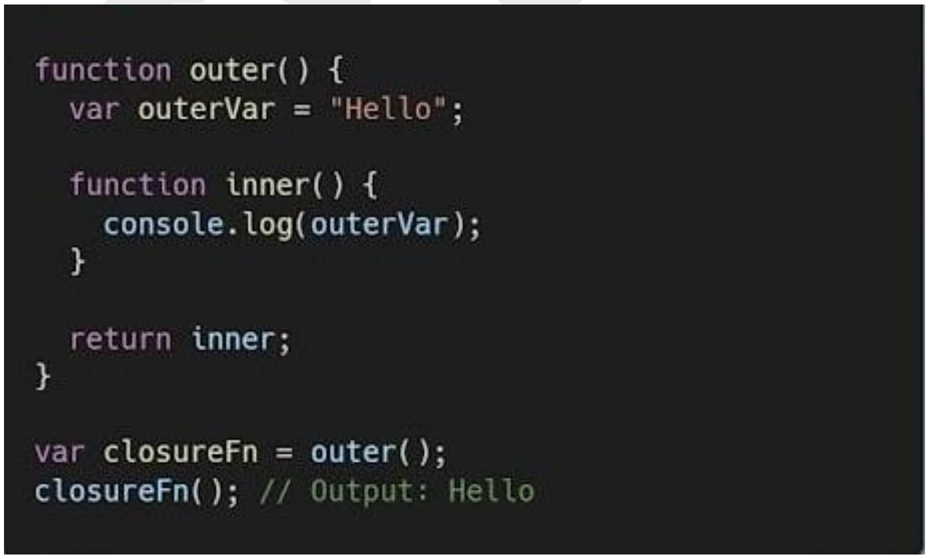
The var keyword declares a global variable, which means that the variable can be accessed from anywhere in the code. The let keyword declares a local variable, which means that the variable can only be accessed within the block of code where it is declared.



```
{
  let x = 10;

  console.log(x); // 10
}
```

9. What Are Closures in JavaScript?

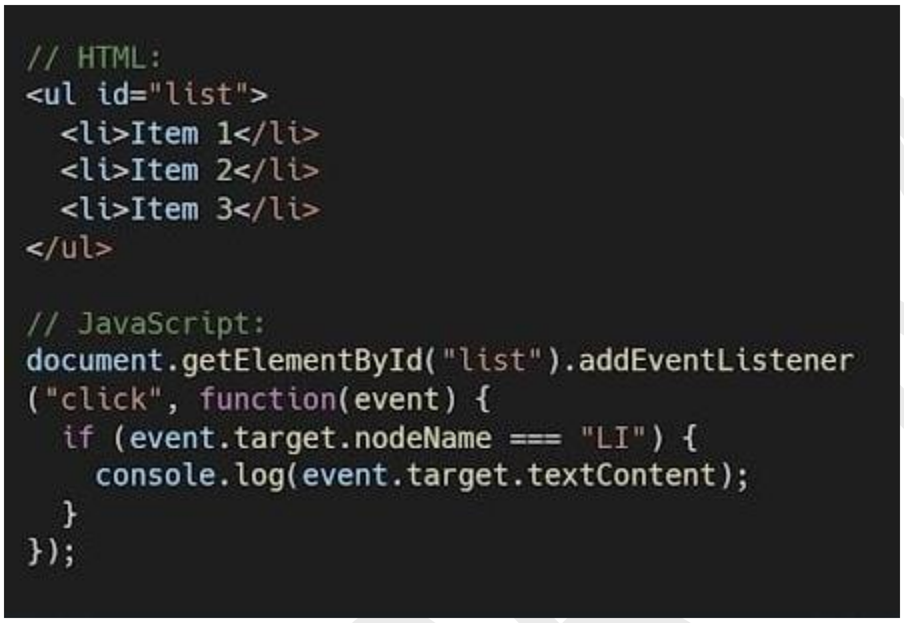
Closures (closureFn) are functions that have access to variables from an outer function even after the outer function has finished executing. They “remember” the environment in which they were created.



```
function outer() {
  var outerVar = "Hello";

  function inner() {
    console.log(outerVar);
  }

  return inner;
}

var closureFn = outer();
closureFn(); // Output: Hello
```

10. What Is Event Delegation in JavaScript?

Event delegation is a technique where you attach a single event listener to a parent element, and that event listener handles events occurring on its child elements. It helps optimize performance and reduce memory consumption.

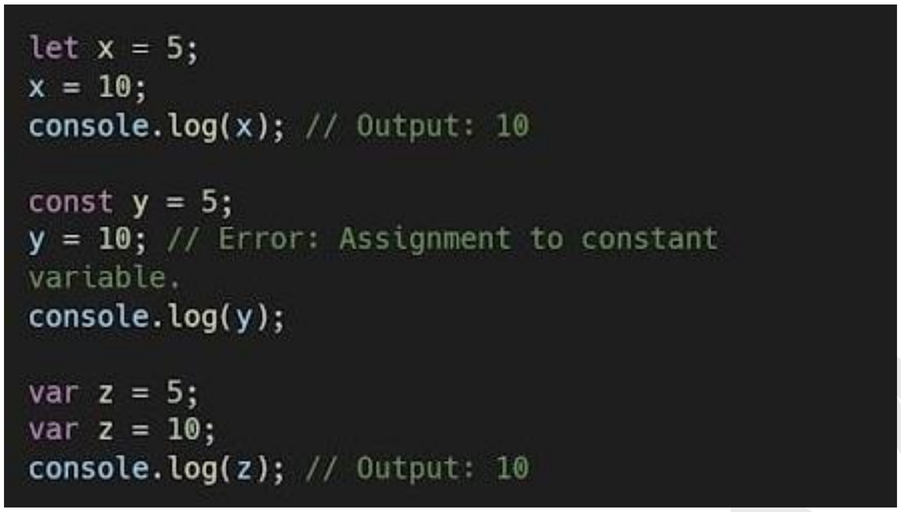


```
// HTML:
<ul id="list">
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
</ul>

// JavaScript:
document.getElementById("list").addEventListener
("click", function(event) {
  if (event.target.nodeName === "LI") {
    console.log(event.target.textContent);
  }
});
```

11. What Is the Difference Between “let”, “const”, and “var”?

let and const were introduced in ES6 and have block scope. let is reassignable, and const is non-reassignable. var is function-scoped and can be redeclared and reassigned throughout the function.

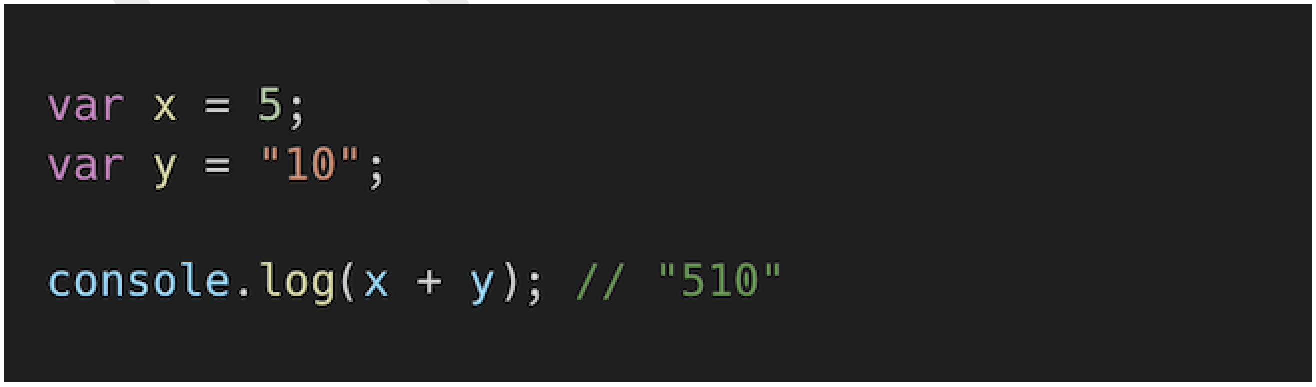
CYBRM


```
let x = 5;
x = 10;
console.log(x); // Output: 10

const y = 5;
y = 10; // Error: Assignment to constant
variable.
console.log(y);

var z = 5;
var z = 10;
console.log(z); // Output: 10
```

12. What Is Implicit Type Conversion in JavaScript?

Implicit type conversion is a JavaScript concept that refers to the automatic conversion of a value from one type to another. In JavaScript, this conversion follows a priority order that typically begins with strings, then numbers, and finally booleans. If you try to add a string to a number, JavaScript will implicitly coerce the number to a string before performing the addition operation because strings have the highest priority in type coercion.

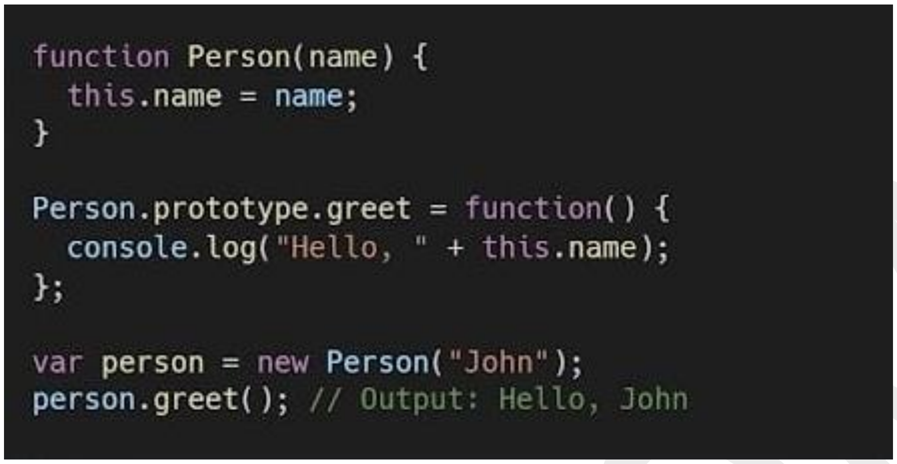
For example, when you combine the number 5 with the string '10' using the addition operator, the result is the string '510'. This occurs because JavaScript will implicitly convert the number 5 to a string following the priority of coercion, and then concatenate it to the string '10'.


```
var x = 5;
var y = "10";

console.log(x + y); // "510"
```

13. Explain the Concept of Prototypes in JavaScript.

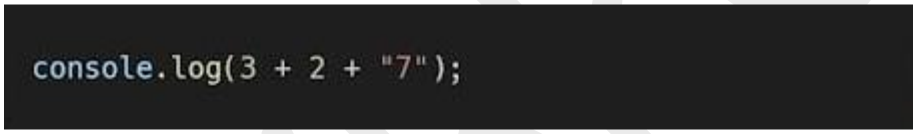
CYBROM

Prototypes are a mechanism used by JavaScript objects for inheritance. Every JavaScript object has a prototype, which provides properties and methods that can be accessed by that object.



```
function Person(name) {
  this.name = name;
}

Person.prototype.greet = function() {
  console.log("Hello, " + this.name);
};

var person = new Person("John");
person.greet(); // Output: Hello, John
```

14. What Is the Output of the Following Code?



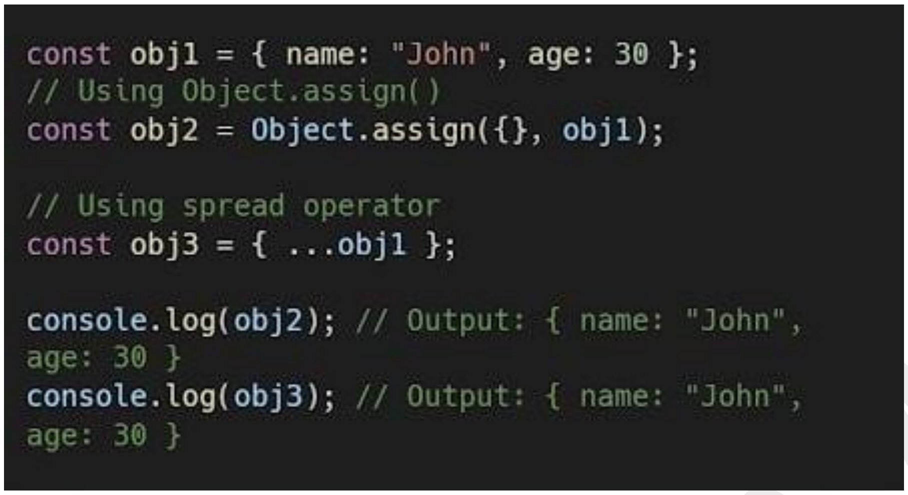
```
console.log(3 + 2 + "7");
```

The output will be "57". The addition operation is performed from left to right, and when a string is encountered, it performs concatenation.

15. How Can You Clone an Object in JavaScript?

There are multiple ways to clone an object in JavaScript. One common method is using the Object.assign() method or the spread operator (...).

CYBROM

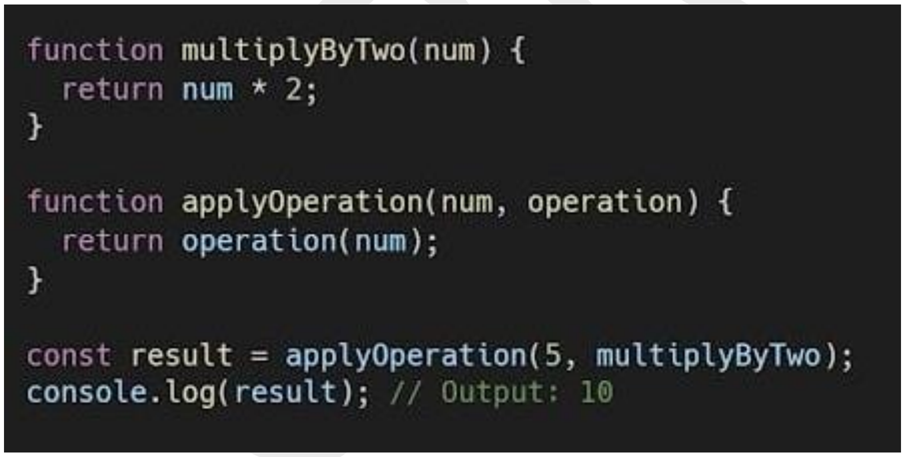


```javascript
const obj1 = { name: "John", age: 30 };
// Using Object.assign()
const obj2 = Object.assign({}, obj1);

// Using spread operator
const obj3 = { ...obj1 };

console.log(obj2); // Output: { name: "John",
age: 30 }
console.log(obj3); // Output: { name: "John",
age: 30 }
```

Intermediate JavaScript Concepts

16. What Are Higher-Order Functions in JavaScript?

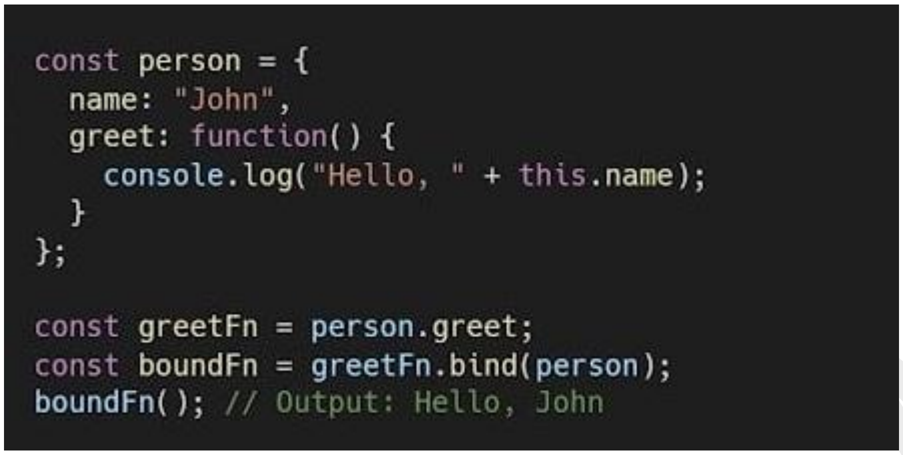
Higher-order functions are functions that can accept other functions as arguments or return functions as their results. They enable powerful functional programming patterns in JavaScript.



```javascript
function multiplyByTwo(num) {
  return num * 2;
}

function applyOperation(num, operation) {
  return operation(num);
}

const result = applyOperation(5, multiplyByTwo);
console.log(result); // Output: 10
```

17. What Is the Purpose of the bind() Method in JavaScript?

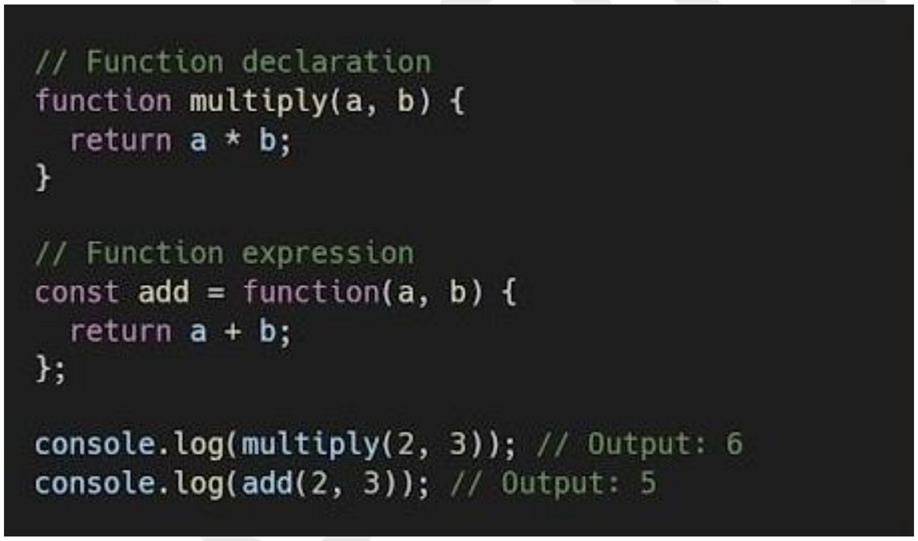
The bind() method is used to create a new function with a specified this value and an initial set of arguments. It allows you to set the context of a function permanently.



```
const person = {
  name: "John",
  greet: function() {
    console.log("Hello, " + this.name);
  }
};

const greetFn = person.greet;
const boundFn = greetFn.bind(person);
boundFn(); // Output: Hello, John
```

18. What Is the Difference Between Function Declarations and Function Expressions?

Function declarations are defined using the function keyword, while function expressions are defined by assigning a function to a variable. Function declarations are hoisted, while function expressions are not.



```
// Function declaration
function multiply(a, b) {
  return a * b;
}

// Function expression
const add = function(a, b) {
  return a + b;
};

console.log(multiply(2, 3)); // Output: 6
console.log(add(2, 3)); // Output: 5
```

19. What Are the Different Types of Errors in JavaScript?

JavaScript can throw a variety of errors, including:

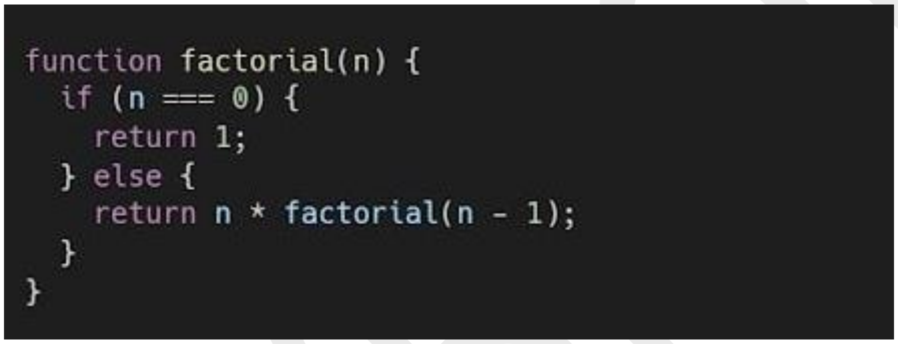
- **Syntax errors**: These errors occur when the JavaScript code is not syntactically correct.

- **Runtime errors**: These errors occur when the JavaScript code is executed and there is a problem.
- **Logical errors**: These errors occur when the JavaScript code does not do what it is supposed to do.

20. What Is Memoization in JavaScript?

Memoization is a technique that can be used to improve the performance of JavaScript code. Memorization works by storing the results of expensive calculations in a cache. This allows the JavaScript code to avoid re-performing the expensive calculations if the same input is provided again.

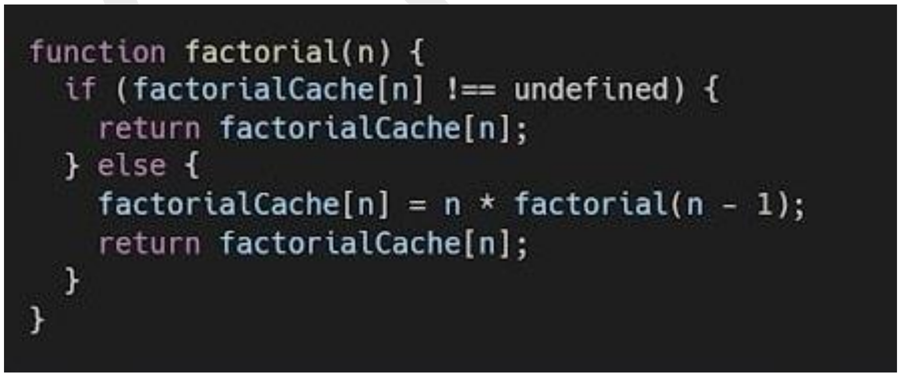
For example, the following code calculates the factorial of a number. The factorial of a number is the product of all the positive integers from one to the number.



```
function factorial(n) {
  if (n === 0) {
    return 1;
  } else {
    return n * factorial(n - 1);
  }
}
```

JavaScript code to calculate the factorial of all positive integers from one to a number

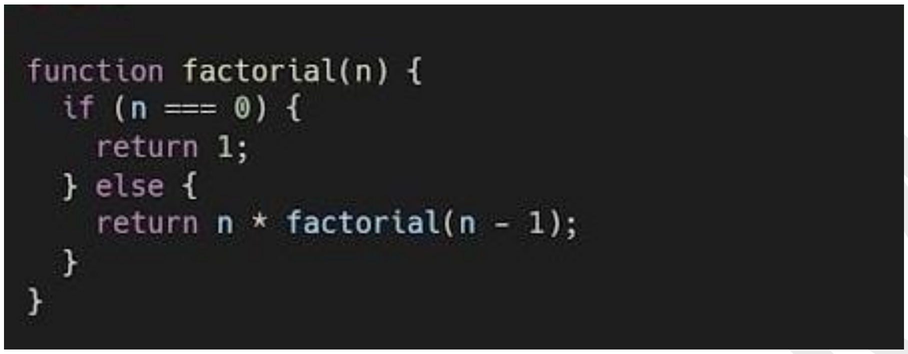
This code can be memoized as follows:



```
function factorial(n) {
  if (factorialCache[n] !== undefined) {
    return factorialCache[n];
  } else {
    factorialCache[n] = n * factorial(n - 1);
    return factorialCache[n];
  }
}
```

21. What Is Recursion in JavaScript?

Recursion is a programming technique that allows a function to call itself. Recursion can be used to solve a variety of problems, such as finding the factorial of a number or calculating the Fibonacci sequence.
The following code shows how to use recursion to calculate the factorial of a number:

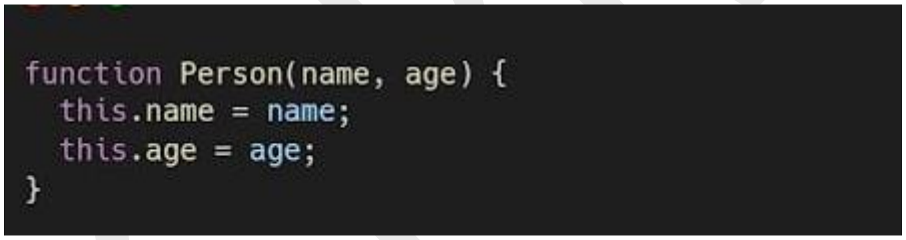


```
function factorial(n) {
  if (n === 0) {
    return 1;
  } else {
    return n * factorial(n - 1);
  }
}
```

22. What Is the Use of a Constructor Function in JavaScript?

A constructor function is a special type of function that is used to create objects. Constructor functions are used to define the properties and methods of an object.

The following code shows how to create a constructor function:

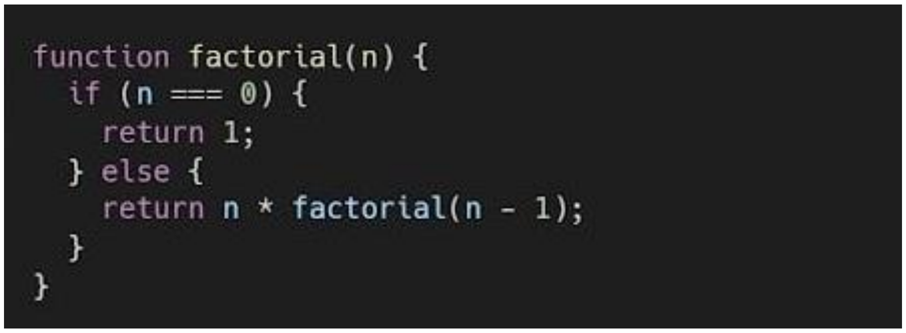


```
function Person(name, age) {
  this.name = name;
  this.age = age;
}
```

23. What Is the Difference Between a Function Declaration and a Function Expression in JavaScript?

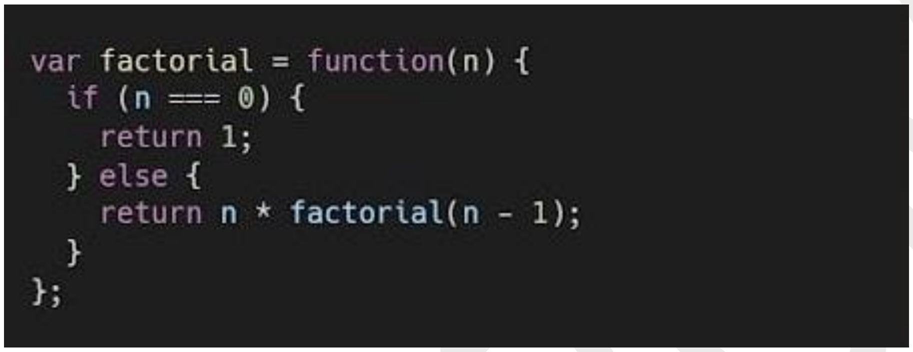
A function declaration is a statement that defines a function. A function expression is an expression that evaluates to a function.

The following code shows an example of a function declaration. This code defines a function named factorial. The factorial function calculates the factorial of a number.



```
function factorial(n) {
  if (n === 0) {
    return 1;
  } else {
    return n * factorial(n - 1);
  }
}
```

The following code shows an example of a function expression:

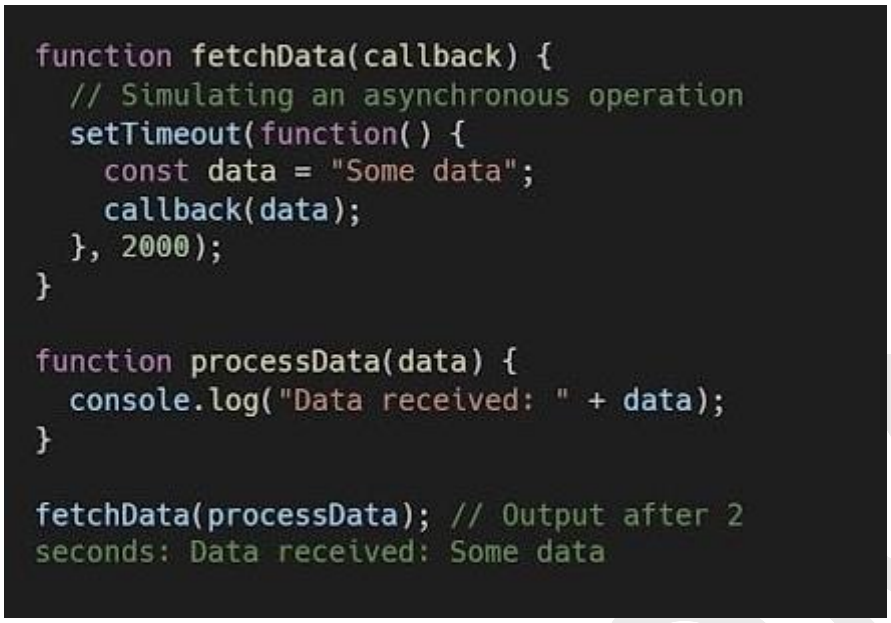


```
var factorial = function(n) {
  if (n === 0) {
    return 1;
  } else {
    return n * factorial(n - 1);
  }
};
```

24. What Is a Callback Function in JavaScript?

A callback function is a function passed as an argument to another function, which is then invoked inside the outer function. It allows asynchronous or event-driven programming.

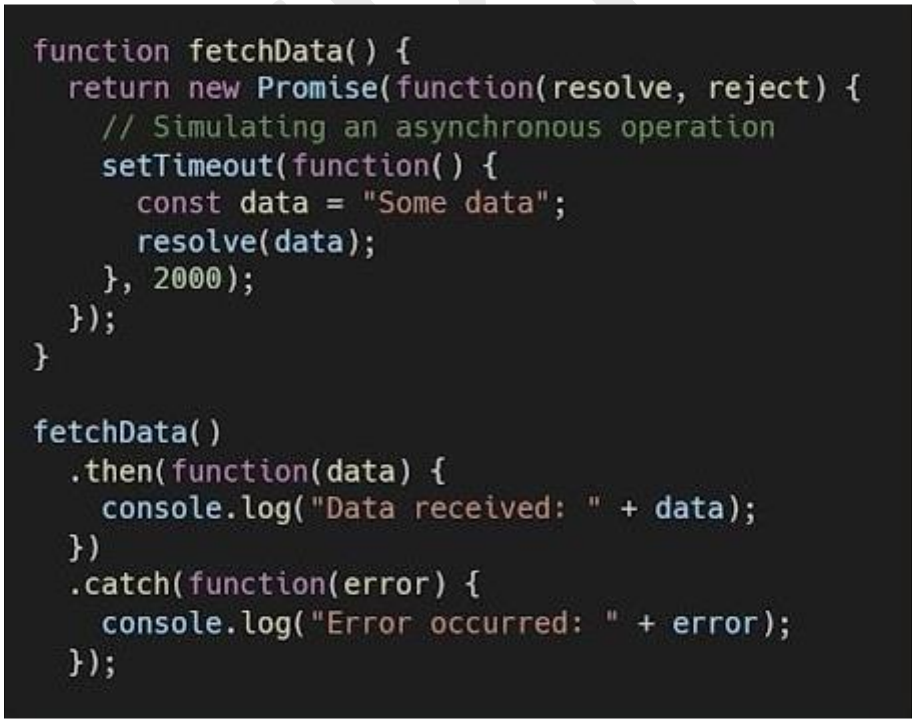
CYBROM



```
function fetchData(callback) {
  // Simulating an asynchronous operation
  setTimeout(function() {
    const data = "Some data";
    callback(data);
  }, 2000);
}

function processData(data) {
  console.log("Data received: " + data);
}

fetchData(processData); // Output after 2
seconds: Data received: Some data
```

25. What Are Promises in JavaScript?

Promises are objects used for asynchronous operations. They represent the eventual completion or failure of an asynchronous operation and allow chaining and handling of success or error cases.



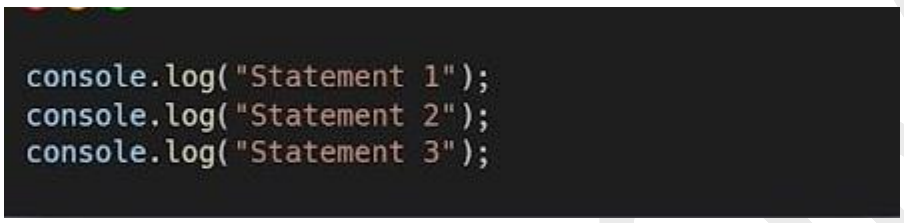
```
function fetchData() {
  return new Promise(function(resolve, reject) {
    // Simulating an asynchronous operation
    setTimeout(function() {
      const data = "Some data";
      resolve(data);
    }, 2000);
  });
}

fetchData()
  .then(function(data) {
    console.log("Data received: " + data);
  })
  .catch(function(error) {
    console.log("Error occurred: " + error);
  });
```

PROM

26. What Is the Difference Between Synchronous and Asynchronous Programming?

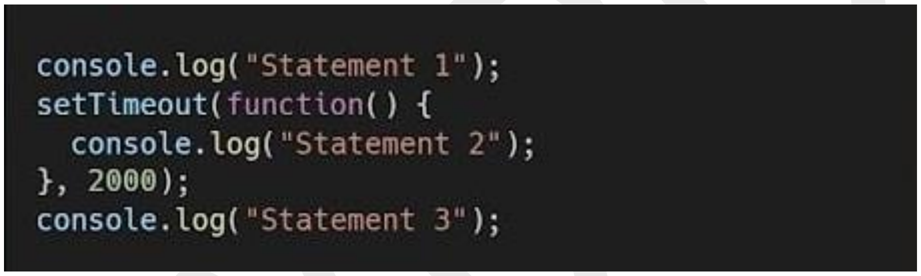
In synchronous programming, the program execution occurs sequentially, and each statement blocks the execution until it is completed. In asynchronous programming, multiple tasks can be executed concurrently, and the program doesn’t wait for a task to finish before moving to the next one.

Synchronous coding example:



```
console.log("Statement 1");
console.log("Statement 2");
console.log("Statement 3");
```

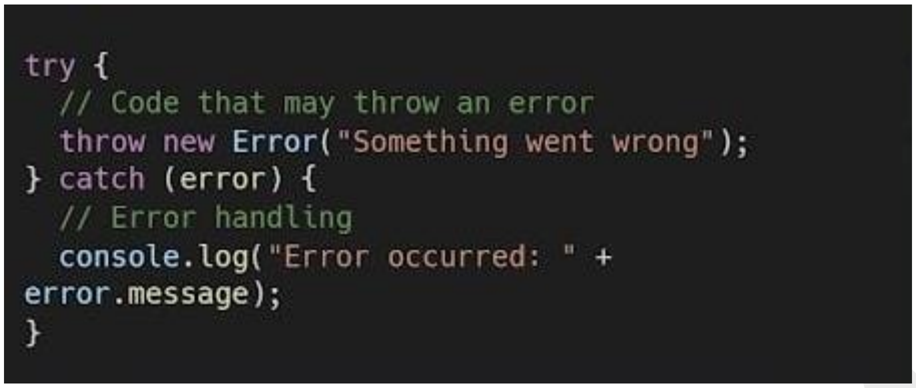
Asynchronous code example:



```
console.log("Statement 1");
setTimeout(function() {
  console.log("Statement 2");
}, 2000);
console.log("Statement 3");
```

27. How Do You Handle Errors in JavaScript?

Errors in JavaScript can be handled using try-catch blocks. The try block contains the code that may throw an error, and the catch block handles the error and provides an alternative execution path.

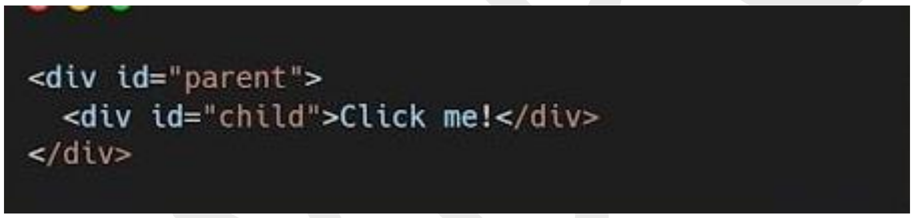


```
try {
  // Code that may throw an error
  throw new Error("Something went wrong");
} catch (error) {
  // Error handling
  console.log("Error occurred: " +
error.message);
}
```

28. Explain the Concept of Event Bubbling in JavaScript.

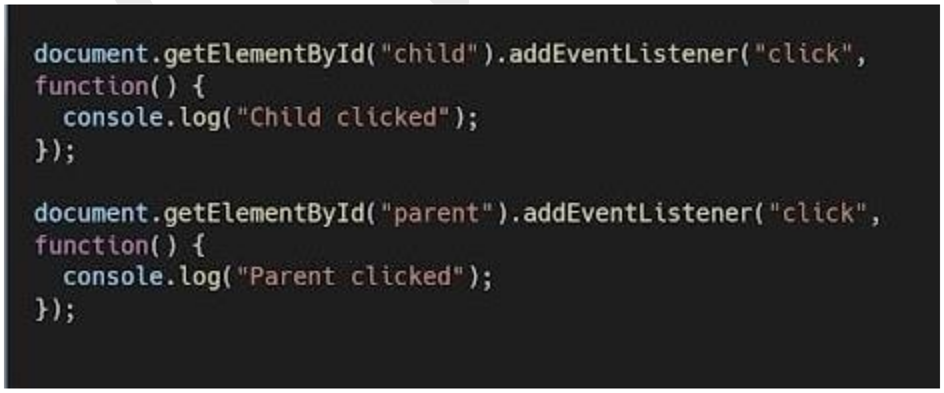
Event bubbling is the process where an event triggers on a nested element, and then the same event is propagated to its parent elements in the document object model (DOM) tree. It starts from the innermost element and goes up to the document root.

Example:



```
<div id="parent">
  <div id="child">Click me!</div>
</div>
```

When you click on the child element, both the child and parent event handlers will be triggered, and the output will be:

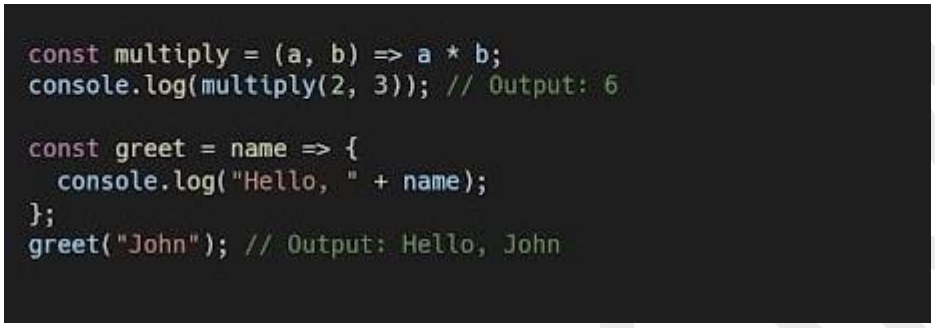


```
document.getElementById("child").addEventListener("click",
function() {
  console.log("Child clicked");
});

document.getElementById("parent").addEventListener("click",
function() {
  console.log("Parent clicked");
});
```

29. What Are Arrow Functions in JavaScript?

Arrow functions are a concise syntax for writing JavaScript functions. They have a more compact syntax compared to traditional function expressions and inherit the this value from their surrounding scope.

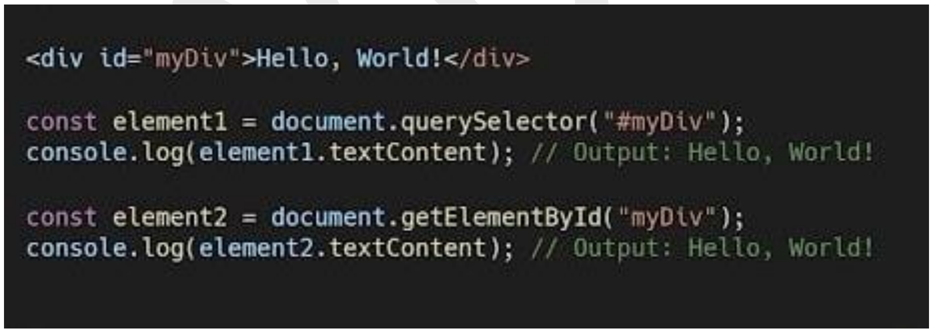
For example:



```
const multiply = (a, b) => a * b;
console.log(multiply(2, 3)); // Output: 6

const greet = name => {
  console.log("Hello, " + name);
};
greet("John"); // Output: Hello, John
```

30. What Is the Difference Between querySelector and getElementById?

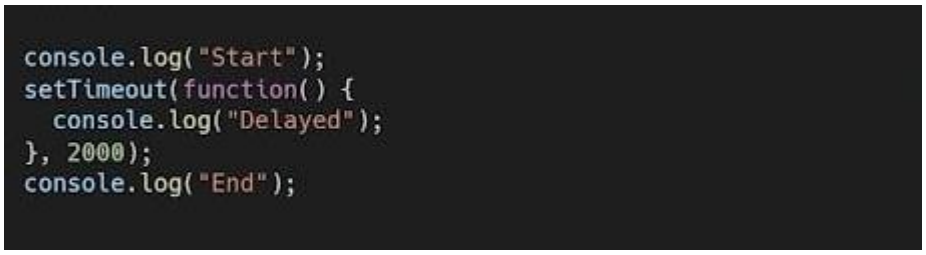
querySelector is a more versatile method that allows you to select elements using CSS-like selectors, while getElementById specifically selects an element with the specified ID.



```
<div id="myDiv">Hello, World!</div>

const element1 = document.querySelector("#myDiv");
console.log(element1.textContent); // Output: Hello, World!

const element2 = document.getElementById("myDiv");
console.log(element2.textContent); // Output: Hello, World!
```

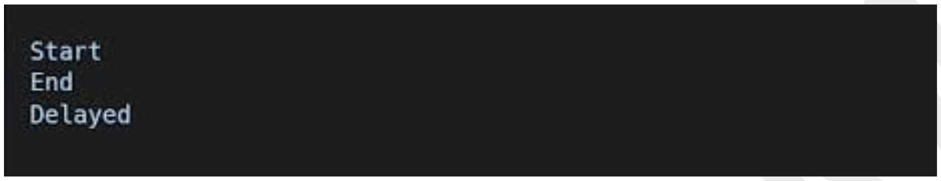
31. What Is the Purpose of the setTimeout() Function in JavaScript?

The setTimeout() function is used to delay the execution of a function or the evaluation of an expression after a specified amount of time in milliseconds.



```
console.log("Start");
setTimeout(function() {
  console.log("Delayed");
}, 2000);
console.log("End");
```

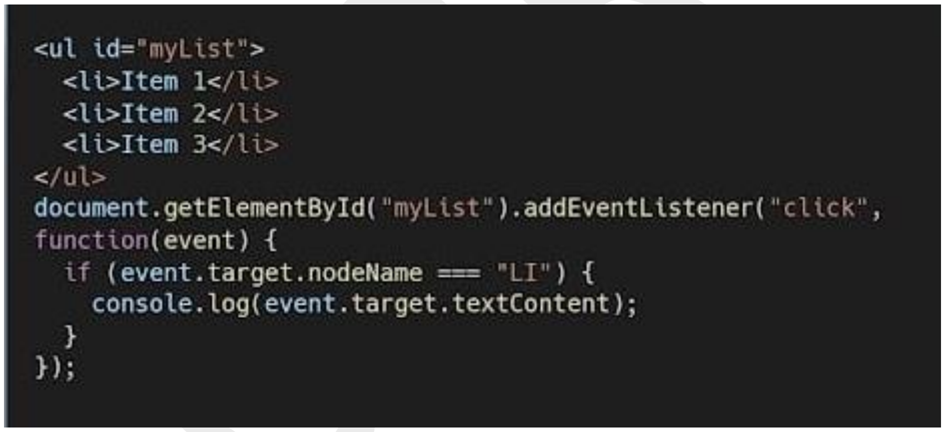
Output after two seconds:



```
Start
End
Delayed
```

32. What Is Event Delegation and Why Is It Useful?

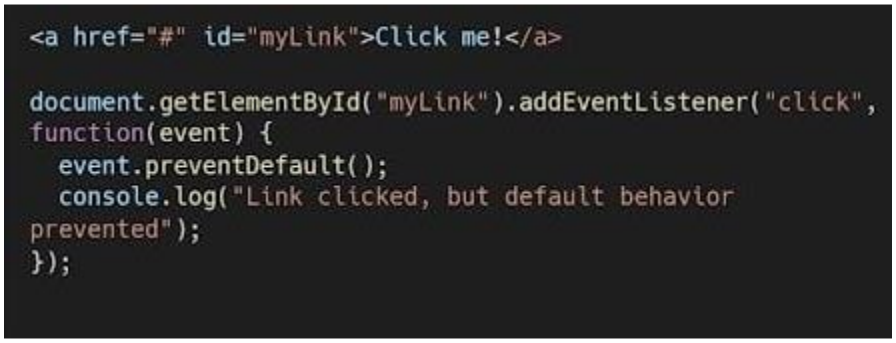
Event delegation is a technique where you attach a single event listener to a parent element to handle events occurring on its child elements. It's useful for dynamically created elements or when you have a large number of elements.



```
<ul id="myList">
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
</ul>
document.getElementById("myList").addEventListener("click",
function(event) {
  if (event.target.nodeName === "LI") {
    console.log(event.target.textContent);
  }
});
```

33. How Can You Prevent the Default Behavior of an Event in JavaScript?

You can use the preventDefault() method on the event object within an event handler to prevent the default behavior associated with that event.

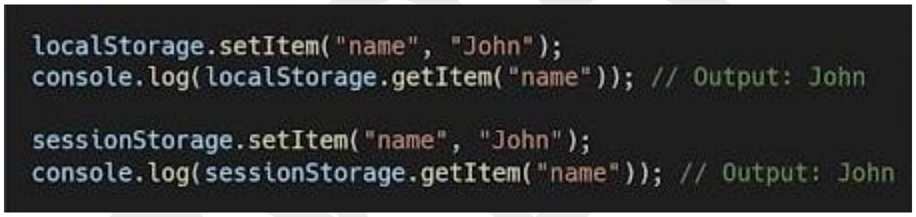


```
<a href="#" id="myLink">Click me!</a>

document.getElementById("myLink").addEventListener("click",
function(event) {
  event.preventDefault();
  console.log("Link clicked, but default behavior
prevented");
});
```

34. What Is the Difference Between localStorage and sessionStorage in JavaScript?

Both localStorage and sessionStorage are web storage objects in JavaScript, but they have different scopes and lifetimes.

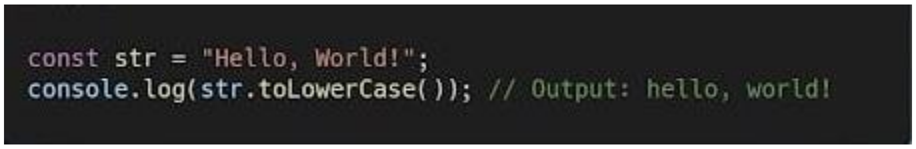
- localStorage persists data even after the browser window is closed and is accessible across different browser tabs/windows of the same origin.
- sessionStorage stores data for a single browser session and is accessible only within the same tab or window.



```
localStorage.setItem("name", "John");
console.log(localStorage.getItem("name")); // Output: John

sessionStorage.setItem("name", "John");
console.log(sessionStorage.getItem("name")); // Output: John
```

35. How Can You Convert a String to Lowercase in JavaScript?

You can use the toLowerCase() method to convert a string to lowercase in JavaScript.

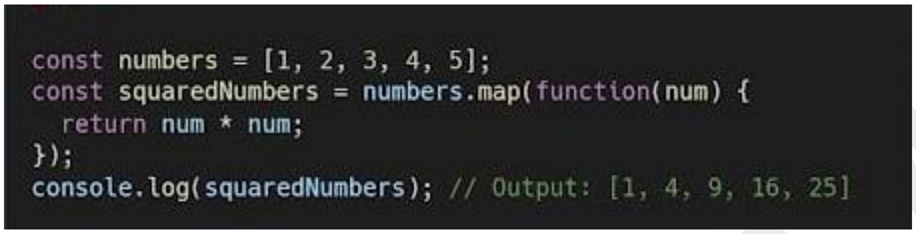


```
const str = "Hello, World!";
console.log(str.toLowerCase()); // Output: hello, world!
```

Advanced JavaScript Concepts

36. What Is the Purpose of the map() Function in JavaScript?

The map() function is used to iterate over an array and apply a transformation or computation on each element. It returns a new array with the results of the transformation.

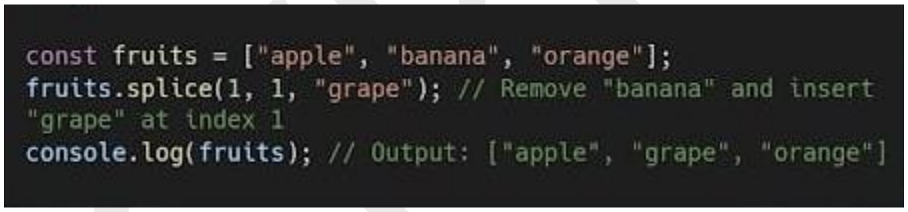


```
const numbers = [1, 2, 3, 4, 5];
const squaredNumbers = numbers.map(function(num) {
  return num * num;
});
console.log(squaredNumbers); // Output: [1, 4, 9, 16, 25]
```

37. What Is the Difference Between splice() and slice()?

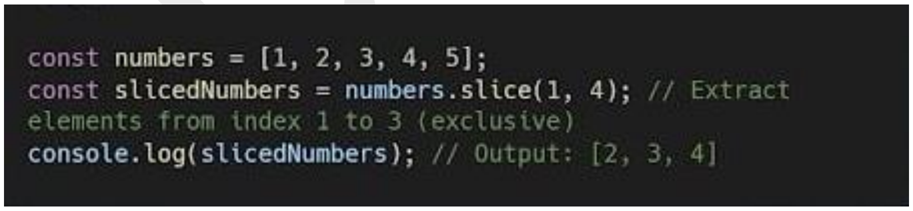
- splice() is used to modify an array by adding, removing, or replacing elements at a specific position.
- slice() is used to create a new array that contains a portion of an existing array, specified by the starting and ending indices.

Example of splice():



```
const fruits = ["apple", "banana", "orange"];
fruits.splice(1, 1, "grape"); // Remove "banana" and insert
"grape" at index 1
console.log(fruits); // Output: ["apple", "grape", "orange"]
```

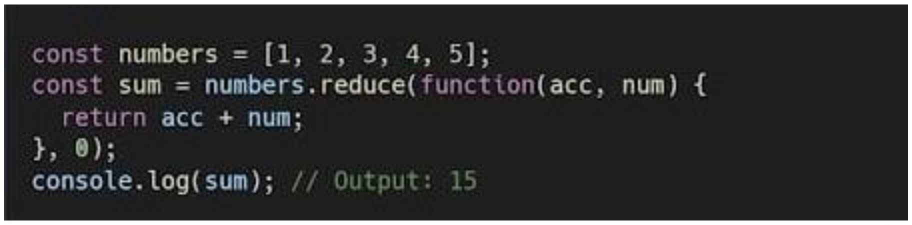
Example of slice():



```
const numbers = [1, 2, 3, 4, 5];
const slicedNumbers = numbers.slice(1, 4); // Extract
elements from index 1 to 3 (exclusive)
console.log(slicedNumbers); // Output: [2, 3, 4]
```

38. What Is the Purpose of the reduce() Function in JavaScript?

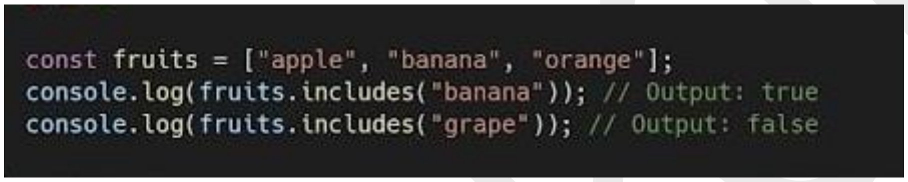
The reduce() function is used to reduce an array to a single value by applying a function to each element and accumulating the result.



```
const numbers = [1, 2, 3, 4, 5];
const sum = numbers.reduce(function(acc, num) {
  return acc + num;
}, 0);
console.log(sum); // Output: 15
```

39. How Can You Check if an Array Includes a Certain Value in JavaScript?

You can use the includes() method to check if an array includes a specific value. It returns true if the value is found, and false otherwise.



```
const fruits = ["apple", "banana", "orange"];
console.log(fruits.includes("banana")); // Output: true
console.log(fruits.includes("grape")); // Output: false
```

40. What Is the Difference Between Prototype and Instance Properties in JavaScript?

A prototype property is a property that is defined on the prototype object of a constructor function. Instance properties are properties that are defined on individual objects that are created by a constructor function.

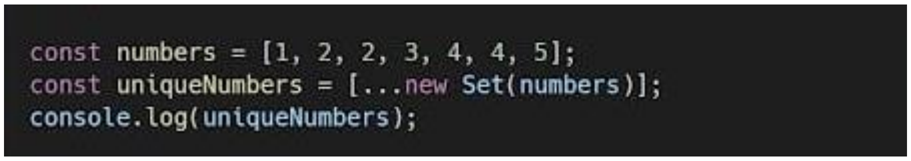
Prototype properties are shared by all objects that are created by a constructor function. Instance properties are not shared by other objects.

41. What Is the Difference Between an Array and an Object in JavaScript?

An array is a data structure that can store a collection of values. An object is a data structure that can store a collection of properties.

CYBROM

Arrays are indexed by numbers. Objects are indexed by strings. Arrays can only store primitive data types and objects. Objects can store primitive data types, objects and arrays.

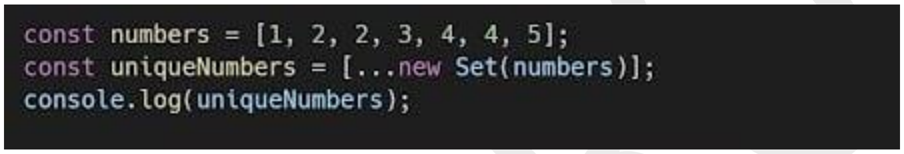


```
const numbers = [1, 2, 2, 3, 4, 4, 5];
const uniqueNumbers = [...new Set(numbers)];
console.log(uniqueNumbers);
```

42. How Can You Remove Duplicates From an Array in JavaScript?

One way to remove duplicates from an array is by using the Set object or by using the filter() method with the indexOf() method.

Example:

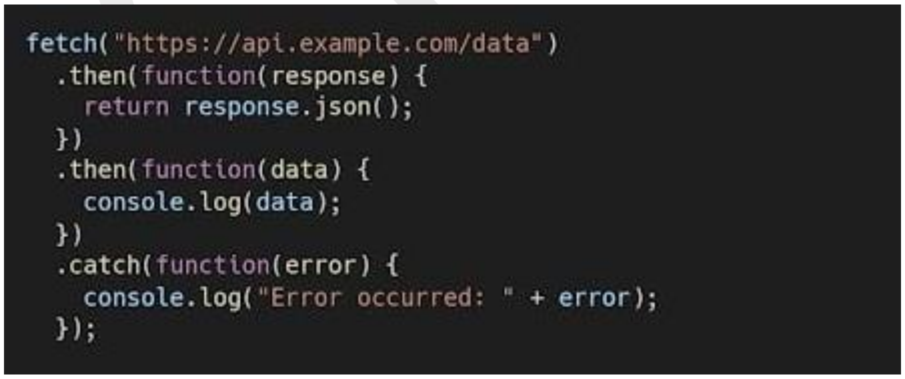


```
const numbers = [1, 2, 2, 3, 4, 4, 5];
const uniqueNumbers = [...new Set(numbers)];
console.log(uniqueNumbers);
```

43. What Is the Purpose of the fetch() function in JavaScript?

The fetch() function is used to make asynchronous HTTP requests in JavaScript. It returns a Promise that resolves to the response from the server.

Example:



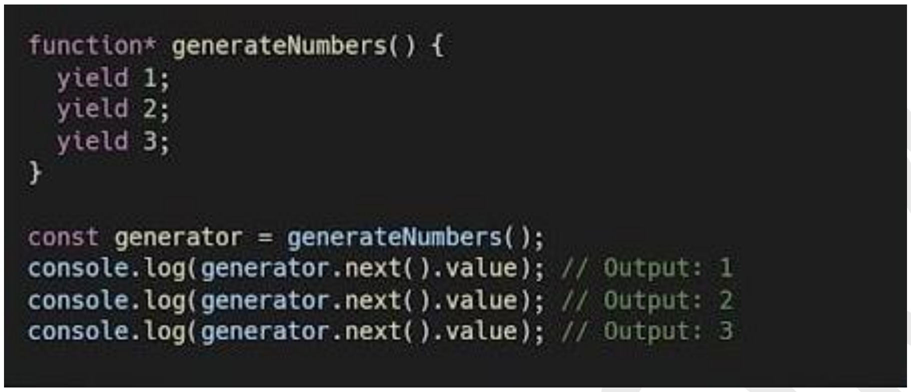
```
fetch("https://api.example.com/data")
  .then(function(response) {
    return response.json();
  })
  .then(function(data) {
    console.log(data);
  })
  .catch(function(error) {
    console.log("Error occurred: " + error);
  });
```

44. What Is a Generator Function in JavaScript?

CYBROM

A generator function is a special type of function that can be paused and resumed during its execution. It allows generating a sequence of values over time, using the yield keyword.

Example:



```
function* generateNumbers() {
  yield 1;
  yield 2;
  yield 3;
}

const generator = generateNumbers();
console.log(generator.next().value); // Output: 1
console.log(generator.next().value); // Output: 2
console.log(generator.next().value); // Output: 3
```

45. What Are the Different Events in JavaScript?

There are many different events in JavaScript, but some of the most common events include:

- Click: The click event occurs when a user clicks on an HTML element.
- Mouseover: The mouseover event occurs when a user's mouse pointer moves over an HTML element.
- Keydown: The keydown event occurs when a user presses a key on the keyboard.
- Keyup: The keyup event occurs when a user releases a key on the keyboard.
- Change: The change event occurs when a user changes the value of an HTML input element.

46. What Are the Different Ways to Access an HTML Element in JavaScript?

There are three main ways to access an HTML element in JavaScript:

1. Using the getElementById() method: The getElementById() method takes a string as an argument and returns the HTML element with the specified ID.

CYBROM

2. Using the getElementsByTagName() method: The getElementsByTagName() method takes a string as an argument and returns an array of all the HTML elements with the specified tag name.
3. Using the querySelector() method: The querySelector() method takes a CSS selector as an argument and returns the first HTML element that matches the selector.

47. What Is the Scope of a Variable in JavaScript?

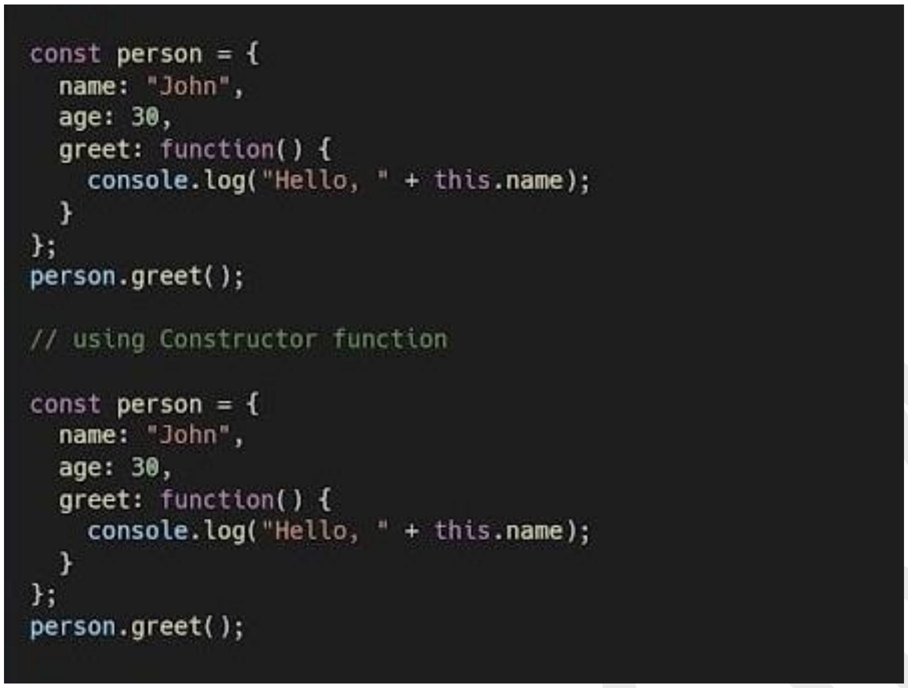
The scope of a variable in JavaScript is the part of the code where the variable can be accessed. Variables declared with the var keyword have a local scope, which means that they can only be accessed within the block of code where they are declared. Variables declared with the let keyword have a block scope, which means that they can only be accessed within the block of code where they are declared and any nested blocks. Variables declared with the const keyword have a global scope, which means that they can be accessed from anywhere in the code.

48. What Are the Different Ways to Create Objects in JavaScript?

There are multiple ways to create objects in JavaScript, including object literals, constructor functions, the Object.create() method and the class syntax introduced in ECMAScript 2015 (ES6).

Example using object literals:

CYBROM



```
const person = {
  name: "John",
  age: 30,
  greet: function() {
    console.log("Hello, " + this.name);
  }
};
person.greet();

// using Constructor function

const person = {
  name: "John",
  age: 30,
  greet: function() {
    console.log("Hello, " + this.name);
  }
};
person.greet();
```

49. What Is the Purpose of the window Object in JavaScript?

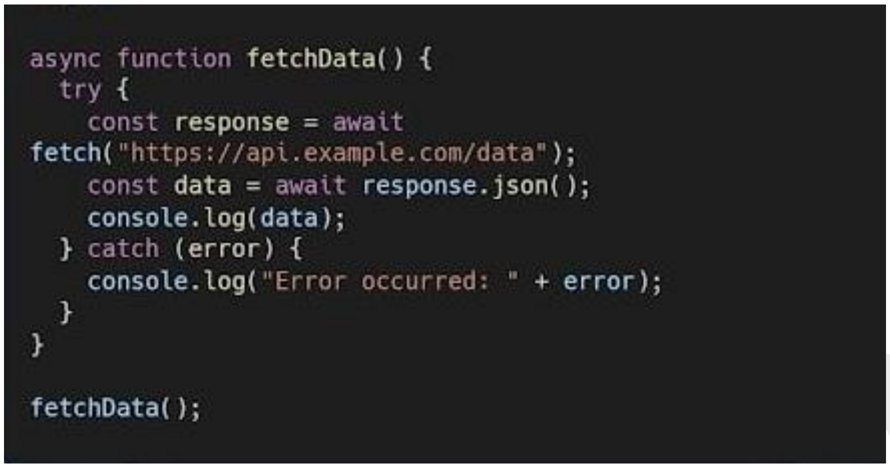
The window object represents the browser window. The window object can be used to access the browser's features, such as the location bar, the status bar and the bookmarks bar.

50. What Is the Purpose of the async and await Keywords in JavaScript?

The async and await keywords are used for handling asynchronous operations in a more synchronous-like manner. The async keyword is used to define an asynchronous function, and the await keyword is used to pause the execution of an async function until a promise is fulfilled or rejected.

Example:

CYBROM



```
async function fetchData() {
  try {
    const response = await
fetch("https://api.example.com/data");
    const data = await response.json();
    console.log(data);
  } catch (error) {
    console.log("Error occurred: " + error);
  }
}

fetchData();
```

51. Why are promises used in JavaScript?

Promises help in managing asynchronous operations, such as server requests in JavaScript. Earlier, callbacks were used for the same purpose. However, callbacks have limited functionality and, thus, can make the code unmanageable. There are four states of the promise object:

Pending: This is the first state of the promise and shows that the promise has been neither fulfilled nor rejected.

Fulfilled: This state represents the completion of the asynchronous operation. In other words, the promise has been fulfilled.

Rejected: This state represents the failure of the asynchronous operation due to some reason. In other words, the promise has been rejected.

Settled: This is the last state of the promise, showing that the promise has been either fulfilled or rejected.

A promise constructor uses a callback with two parameters - resolve and reject - to create a promise. The resolve function is called when the asynchronous operation has succeeded. The reject function is called when the asynchronous operation has failed.

52. What are the different data types present in JavaScript?

CYBROM

There are three major data types present in JavaScript.

Primitive

- Numbers
- Strings
- Boolean

Composite

- Objects
- Functions
- Arrays

Trivial

- Null
- Undefined

53. Why are Arrow functions used almost everywhere?

Arrow functions are used everywhere because:

- Safety of scope: When the arrow function is used everywhere, it brings consistency of scope because the same thisObject is used everywhere. If by any chance, a standard function is used alongside the arrow function, there are chances of the scope getting mixed up.
- Compactness: As compared to the standard function, the arrow function is compact as it does away with the need for keywords, curly braces, parenthesis, etc. in certain cases. It is, therefore, easier to read and write.
- Clarity: When the arrow function is used everywhere, there is a certain consistency of scope. Thus, whenever a standard function is mixed with it, it stands out distinctly. The developer can therefore look for the next higher function to locate the thisObject.

54. Is JavaScript a dynamically typed or statically typed language?

JavaScript is a dynamically typed language.

CYBROM

55. Describe Arrow functions.

The ES6 Javascript version introduced Arrow functions. With the Arrow functions, we can declare functions using new and shorter syntax. These functions can only be used as function expressions. The declaration of these functions is done without using the function keyword. Moreover, if there is a single returning expression, then even the return keyword is not needed. Additionally, wherever the code occurs in a single line only, we can omit the curly {} braces. If there is only one argument in a function, then we can omit even the () parenthesis.

56. Which company developed JavaScript?

The language was developed by Brenden Eich in 1995 when he was working as an employee of netscape. He is also the founder of Mozilla foundation.

57. What are classes in Javascript?

Classes in Javascript are templates for building objects. Classes bind the data with code so that the data works as per the code. They were introduced in the ES6 version of Javascript and while they were created on prototypes, they also have syntax and semantics that are not common with ES5. Classes can be seen as special functions. There are two components of class syntax: class expressions and class declarations.

Class expressions are one of the ways to define classes. They may or may not have names. If a class expression has a name, it is held locally in the class body but can be accessed through the name property. Before using class expressions, one must declare them.

Another way to define classes is class declaration. For declaring a class, the class keyword must be followed by the class name.

One class may use the properties of methods of another class by using the extend keyword. Classes in Javascript must follow the strict mode. If the strict mode is not followed, errors will appear.

58. Explain rest parameter and spread operator.

The ES6 version of Javascript introduced the rest parameter and spread operator.

CYBROM

Rest Parameter

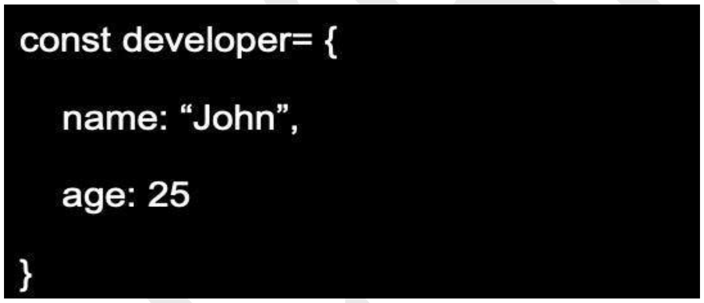
The use of three dots (...) before any parameter shows a rest parameter. This improves the handling of function parameters. With the help of the rest parameter, we can create such functions that can take a different number of arguments. By using the rest parameter, all these arguments will be converted into arrays. The rest parameter can also be used to extract any or all parts of an argument.

Spread Operator

Though the spread operator syntax is the same as that of the rest parameter, the spread operators help to spread arrays and object literals. Another use of spread operators is when one or more arguments are expected to be in a given function call. So, while the rest parameter creates an array, a spread operator spreads an array.

59. How can you create objects in JavaScript?

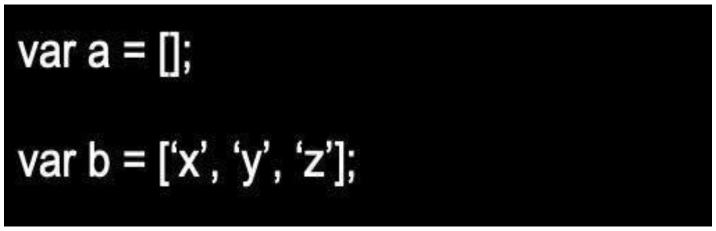
Because JavaScript is fundamentally an object-oriented scripting language, it encourages and supports using objects when developing web-based applications.



```
const developer= {
    name: “John”,
    age: 25
}
```

60. How can you make arrays in JavaScript?

Here's a simple method to create arrays with JavaScript using the array literal:



```
var a = [];

var b = ['x', 'y', 'z'];
```

61. What is object destructuring?

Object destructuring is a JavaScript feature to extract object properties and bind these properties to variables. It can be used to extract several properties in a single statement and can reach properties from nested objects. When no property exists, object destructuring can set default values.

62. Which is faster, JavaScript or ASP script?

JavaScript is faster than ASP script, as it is a lightweight language that is designed to run quickly in the browser. However, ASP script can perform more complex tasks that JavaScript cannot, so it can be faster in certain situations.

63. Explain temporal dead zone.

Temporal dead zone (TDZ) was introduced for the first time in the ES6 version of JavaScript. TDZ is a behavior that occurs when a variable is accessed before it is initialized. It is a language construct that helps to catch errors as accessing data before initializing it is incorrect. When let and const keywords are used to declare variables a TDZ may occur.

64. What is WeakMap in JavaScript?

WeakMap object stores key-value pairs with weakly referenced keys. Keys in WeakMap must only be objects, while values can be arbitrary. Primitive data types cannot be keys in WeakMap. Since native WeakMap contains weakly referenced keys, these keys can be garbage collected and therefore references get removed. Also, because of the weak referencing, garbage collection of values in WeakMap is not prevented. When information about a key is valuable 'only if' the key is not garbage collected, WeakMap is useful in mapping keys to the information about them.

65. What is WeakSet in Javascript?

CYBROM

WeakSet in Javascript is a collection of unique and orderly objects. Unlike Set, a WeakSet does not contain any other elements. Those objects of a collection that are weakly held, appear in WeakSet. This also means that if there is no reference for a weakly held object, it will be collected as garbage. There are three methods of WeakSet: add(), delete(), and has().

66. How to debug JavaScript code?

Modern web browsers, such as Chrome, Firefox, etc., include a built-in debugger that can be opened anytime by pressing the appropriate key, typically the key F12.

67. Which character is used to split JavaScript Code spanning into multiple lines?

The backslash, '' character, is used at the end of the first line if the JavaScript code is spread across multiple lines.

68. What are the undeclared and undefined variables in JavaScript?

Undeclared variables are variables that do not exist in the program and therefore are not declared. If a program attempts to determine the values of an undefined variable, it will fail because of a runtime error.

Undefined variables refer to ones declared by the program but not given a value. If the program attempts to find the values of an undefined variable the variable is returned with an undefined value.

69. Explain global variables in JavaScript.

A variable declared outside of a function definition is referred to as a global variable, and its scope is across your entire program. This implies that its value is accessible and adjustable throughout your program.

70. What does the prompt box mean in JavaScript?

It displays an interactive dialog box that displays an optional prompt for users to enter some text. It is typically used when users want to enter a number before entering a webpage. This returns either a string containing the user's input text or a null.

CYBROM

71. What is NULL in JavaScript?
The NULL value signifies no value or no object. It is also known as an empty value or empty object.

72. What is the "this" keyword in JavaScript?
"this" refers to an object running the current line of code. It is a reference to the object which executes the current function. If the function that is being referenced is a regular one, "this" references the global object. If the function is a method of the object, "this" refers to the object itself.

73. Define the purpose of timers in JavaScript.
Timers can be used to execute the code at an exact time or repeat the code within an interval. This can be accomplished by employing the methods setTimeout, setInterval, and clearInterval.

74. Which character is used to denote a comment?
The // character is for single-line comments, and the /* */ characters are used for multi-line comments.

75. Differentiate between ViewState and SessionState?

ViewState: It is specific to the page state within the browser in a session.

SessionState: It's user-specific containing the data of the user session and allows access to all data on the web pages.

76. What is the use of the === operator?
The === operator is a strict comparison operator, meaning it checks for both the value and the type of two variables.

77. Does JavaScript support automatic type conversion?
JavaScript supports automatic type conversion. This is the most common method of type conversion that JavaScript developers use.

78. Define Variable Typing in JavaScript.

CYBROM

Variable typing allows you to assign a numerical value to a variable. The same variable could be assigned to a string.

79. What is the main difference between "==" and "===" in JavaScript?

== tests only for value equality; however, "===" is a stricter equality test that returns false if the value or the type of two variables is different.

80. How to find the operating system in the client machine using JavaScript?

To detect the operating system, we can use a navigator.appVersion or navigator.userAgent property in JavaScript.

81. What's the purpose of the delete operator in JavaScript?

Delete operator is used to delete the property and its value.

82. List all Pop-up box types in JavaScript.

The pop-up boxes available in JavaScript are Alert, Confirm, and Prompt.

83. What is the use of Void(0)?

Void(0) stops pages from refreshing, and "zero" is used to pass the parameter while calling.

84. Differentiate between an alert box and a confirmation box.

An alert box shows only one button, an OK button. A confirmation box has two buttons: the OK and Cancel buttons.

85. Define escape characters.

In JavaScript, escape characters are used to represent special characters within a string. Escape characters start with a backslash followed by a character. The character that follows the backslash determines the special character being represented.

86. Explain JavaScript Cookies.

CYBROM

Cookies are small text files saved on the computer. They are created to store the required information when users browse the website. Examples include Users Name information and information about shopping carts from previous visits.

87. Explain the purpose of the pop() method.

This pop() method works like the shift() method. However, the difference is the shift method operates from the beginning of the array. The pop() method takes the final element off of the given array and returns it. The array upon which it is called is then altered.

88. What are break and continue statements?

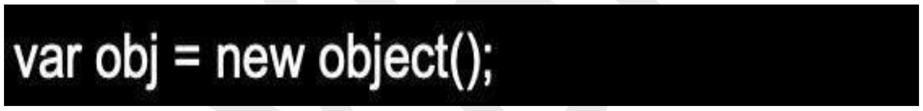
A break statement is a way to exit the current loop. The continue statement continues with the next statement of the loop.

89. What's the purpose of the blur function in JavaScript?

The blur function removes the focus from the specified object.

90. How to create Generic objects in JavaScript?

Generic objects can be created by:



```
var obj = new object();
```

91. What is the outcome of 4+2+"8"?

The 4 and 2, in this case, behave as integers, and "8" behaves like a string. Therefore 4 + 2 equals 6. The output is 6+"8" = 68.

92. What keywords are used to handle the exceptions?

The keywords commonly used to handle exceptions are: try, catch, finally, throw, and throws.

93. How to print screen in JavaScript?

By calling the window.print() method in the browser, we can print the content of the current window in JavaScript.

CYBROM

94. How to submit a form using JavaScript?

By using document.form[0].submit(), you can submit a form using JavaScript.

95. How to read a cookie?

The process of reading a cookie with JavaScript is also very easy. We can use the document.cookie string to store the cookies we have just created with the string.

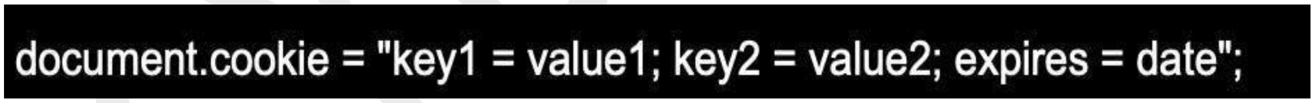
The document.cookie string stores a list of name-value pairs separated by semicolons. The "name" is the name given to the cookie, and the "value" is its value. You can also employ split() method to split the value of the cookie into values and keys.

96. How to delete a cookie using JavaScript?

If you'd like to delete a cookie, so that future attempts to read it in JavaScript return nothing, then you must change the date of expiration to a date in the past. Defining the cookie's path is important to ensure you erase the correct cookie. Certain browsers won't allow you to delete a cookie if the path is not specified.

97. How to create cookies using JavaScript?

To create cookies using JavaScript, you must assign a string value to the document.cookie object.



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

98. What is the main difference between Attributes and Property?

Attributes- It provides more details on an element like id, type, value, etc.

Property- The value assigned to the property such as type="text" value='Name'.

99. How an HTML element is accessible in JavaScript code?

Here are the methods by which an HTML element can be used in JavaScript code:

CYBROM

getElementById('idname'): Gets an element by its ID name

getElementsByClass('classname'): Gets all the elements that have the given class name.

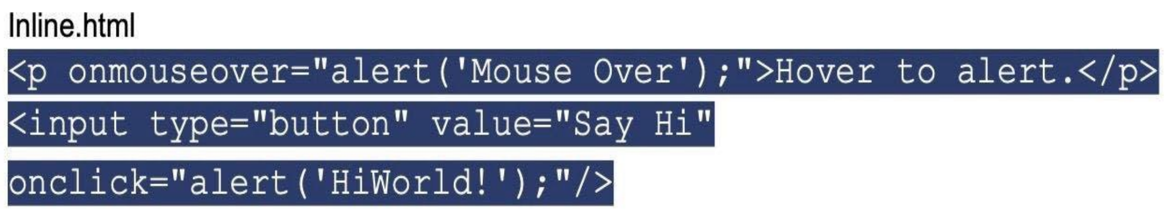
getElementsByTagName('tagname'): Gets all the elements with the given tag name.

querySelector(): It returns the first selected element using a CSS style selector.

100. In what ways a JavaScript code can be involved in an HTML file?

There are three ways that a JavaScript code can be used in the HTML file: Inline, Internal, and External.

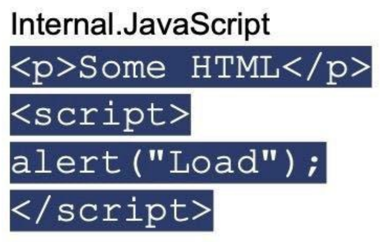
Inline JavaScript



Inline.html

```
<p onmouseover="alert('Mouse Over');">Hover to alert.</p>
<input type="button" value="Say Hi"
onclick="alert('HiWorld!');"/>
```

Internal JavaScript



Internal.JavaScript

```
<p>Some HTML</p>
<script>
alert("Load");
</script>
```

External JavaScript

1a-external.js

```
alert("1A Load");
```

1b-external.html

```
<!-- (A) RELATIVE URL -->

<script src="1a-external.js"></script>

<!-- (B) ABSOLUTE URL -->
<script src="http://localhost/1a-external.js"></script>
```

101. What is unshift method in JavaScript?

It's used to add elements to the front of the array and provides the updated length of the array. It's similar to a push method, which inserts elements at the beginning of the array.

102. What is a Typed language?

Typed Language is where the datatype is defined which is interpreted by the machine at runtime.The typed language can be classified into two types:

Dynamically: In this case, the variable could contain multiple types; just like in JS variables, a variable can be a number or characters.

Statically: In this case, the variable can only hold one data type. Like Java, the variable declared of string can hold only one set of characters.

103. Name some JavaScript frameworks.

JavaScript framework is an application framework written using JavaScript. In its control flow, it is different from a JavaScript Library. There are many JavaScript frameworks, but the most popular are: Angular, Node, Bootstrap, and Vue among others.

104. What is the difference between Local storage & Session storage?

Local Storage: The data is not transmitted directly to the server with each HTTP request (HTML, images, JavaScript, CSS, etc), thus reducing the traffic between the server and the client. It remains until removed manually through settings or software.

Session Storage: It is identical to local storage; the only difference is that while the data stored locally does not expire, whereas in session storage, it is deleted when the session is over. Session Storage is removed once the browser closes.

105. What's the difference between a window and a document?

JavaScript window can be described as a global object that contains properties and methods that interact with it.

The document is a part of the window and is considered as the property of the window.

106. What's the distinct difference between innerText and innerHTML?

innerHTML - innertHTML returns the entire text, including the HTML tags, that are contained within an element.

innerText - innerText returning the entire text that is contained within an element as well as its child elements.

107. How to convert the string of a base to an integer using JavaScript?

This parseInt() function can convert numbers between different bases. This function returns an integer of a base specified in the second argument of the parseInt() method.

108. What are Exports and Imports?

Imports and exports allow the writing of modular JavaScript code. With the help of exports and imports, we can break the code into multiple files.

CYBROM

109. What is the 'Strict' Mode in JavaScript?

Strict Mode in JavaScript is a way to opt into a restricted variant of the language, which enforces stricter parsing and error handling rules. It helps developers write more secure and efficient code by catching common mistakes and potential bugs that might otherwise go unnoticed.

When enabled, Strict Mode may throw errors for code that previously worked without any issues, but it does so in order to catch potential issues that could lead to bugs or security vulnerabilities. Strict Mode also disables some features of JavaScript that could be problematic or confusing, such as the automatic creation of global variables.

Overall, Strict Mode is a useful tool for writing safer and more reliable JavaScript code, but it requires developers' care and attention to use effectively.

110. Demonstrate the importance of isNaN() function in JavaScript.

The isNaN() function is used to determine whether the given value is an illegal number or not. If it returns true, the value is a NaN - Not a number. Else returns false. It differs from the Number.isNaN() method.

111. What are the types of errors found in JavaScript?

JavaScript has three types of errors: Runtime, Logical, and Syntax.

Syntax errors- These occur when the code violates the rules of the JavaScript language. For example, a missing semicolon, an unclosed string, or an unexpected token.

Logical errors - These occur when the code does not produce the desired outcome, but there is no error message displayed. Logical errors are usually caused by mistake in the code's logic or algorithm.

Runtime errors- These occur when the code is syntactically correct and logically sound, but still throws an error during execution. For example, accessing an undefined variable, calling a function that does not exist, or attempting to divide by zero.

CYBROM

112. How would you go about debugging a JavaScript program that is not running correctly?

The first step to debugging a JavaScript program is to identify the error. This can be done by using the JavaScript console in the browser’s developer tools. The console will display any errors that occur during script execution such as syntax errors, type errors, and reference errors. Once the error has been identified, the code can be examined line by line to find the source of the issue and the issue can be addressed.

113. Which method extracts a character from a certain index?

We can retrieve a character from a certain index using the charAt() function.

114. What is the design prototype pattern?

The Prototype design Pattern is described as a property pattern or prototype pattern that can be used to create different objects and prototypes that are copied using a template with specific values.
The key idea behind the Prototype Pattern is to use an existing object as a template and create new instances by copying its properties. This approach can be more efficient than creating new instances from scratch, especially if the object has a complex initialization process. By copying an existing object, we can avoid the need to repeat the initialization process for every new object that we create.

115. How is the Temporal Dead Zone defined?

AIt is an area of block where a variable cannot be accessed until it is completely initialized with a value by the computer. Incase, one tries to access it before complete initialization, JavaScript throws “ReferenceError”. It is defined using const and let keywords.

116. What is the difference between the methods .map() and .forEach() in JavaScript?

The .map() and .forEach() methods are both used to iterate over an array, but the .map() method can be used to transform an array by returning a new array with the same number of elements but with different values. The .forEach() method is

CYBROM

used to perform an action on each element in the array without returning a new array. For example:

```
// Using the map() method
let arr = [1,2,3];
let newArr = arr.map(element => element * 2);

console.log(newArr); // Outputs [2,4,6]

// Using the forEach() method
let arr = [1,2,3];
let newArr = [];
arr.forEach(element => newArr.push(element * 2));

console.log(newArr); // Outputs [2,4,6]
```

117. How would you go about creating a method that returns the sum of all the elements in an array?

To create a method that returns the sum of all the elements in an array, you can use the Array.prototype.reduce() method. For example:

```
let arr = [1,2,3,4];
let result = arr.reduce((acc, val) => acc + val, 0);
console.log(result); // Output 10
```

118. How to change style/class of an element?

The style/class of an element can be changed in two ways.

document.getElementById("myText").style.fontSize = "14px;

document.getElementById("myText").className = "anyclass";

119. How can you get rid of duplicates from a JavaScript array?

There are two ways we can eliminate duplicates from the JavaScript array:

Through the Filter Method:

The filter() method can use three arguments: arrays, current element, and the current element's index.

Utilizing the For Loop:

The For loop involves iterating through the original array and checking whether each element has already been added to a new array. If it hasn't, the element is added; if it has, it is skipped.

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

For reading operations, the readFile() function is used.

readFile(Path, Options, Callback)

For writing operations,The writeFile() functions is used.

writeFile(Path, Data, Callback)

121. How can you target a particular frame from a hyperlink in JavaScript?

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

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

122. What is the main difference between JavaScript and JScript?

JavaScript:

- It is a scripting programming language created by Netscape.
- This is utilized to develop the server and client-side software.
- It is independent of Java language.

JScript:

- This is a scripting language created by Microsoft.
- It's used to create interactive online content.

CYBROM

123. How can we hide JavaScript code from old browsers that don't support JavaScript?

To hide the JavaScript codes from old browsers that do not have support for JavaScript, you can make use of:

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

The majority of browsers from the past will consider it as a long comment of HTML. New browsers that support JavaScript will treat it as an online message. Alternatively, you could use the "noscript" element to provide alternative content that will display for users who do not have JavaScript enabled.

124. Can you explain the usage of Internal and External JavaScript Code?

When there are only a few lines of code for a particular webpage, it is preferable to keep JavaScript code internal within the HTML document.
However, if the JavaScript code is used on multiple web pages, it is advisable to keep the code in a separate file.

125. What do you know about unescape() and escape() functions?

The escape () function codes a string to transfer the information across a network from one system to another.

The unescape() function decodes the coded string.

126. Can you explain what screen objects are?

Screen objects read the information from the screen of the client. Some properties of screen objects are

- AvailHeight: Provides the height of the screen
- AvailWidth: Provides the width of the screen
- ColorDepth: Provides the bit depth of images on the screen

CYBROM

- Height: Provides the total height of the screen along with the taskbar
- Width: Provides the total width of the screen, including the taskbar

127. Explain different functional components in JavaScript?

The functional components in JavaScript are-

First-class functions: functions are used as first-class objects. This means they can be passed as arguments to other functions, sent back as values from other functions, assigned to variables, or can even be saved as data structures.

Nested functions: These are defined inside other functions and called every time the main function is called.

128. Explain deferred scripts in JavaScript.

When the page loads, HTML code parsing is paused by default till the time the script has not fully executed. this happens when the server is slow or the script is heavy, and the web page gets delayed. So by using Deferred script , scripts execution is delayed till the HTML parser rums. this helps reduce the loading time of web pages and they can be displayed faster.

129. Explain how event handlers are used in JavaScript.

Whenever the user clicks the link or fills out a form, events take place. Events are the actions that come off as a result of activities. An event handler serves to manage the proper execution of all these events. They are an extra attribute if the object which includes the vent's name and action is taken when the event takes place.

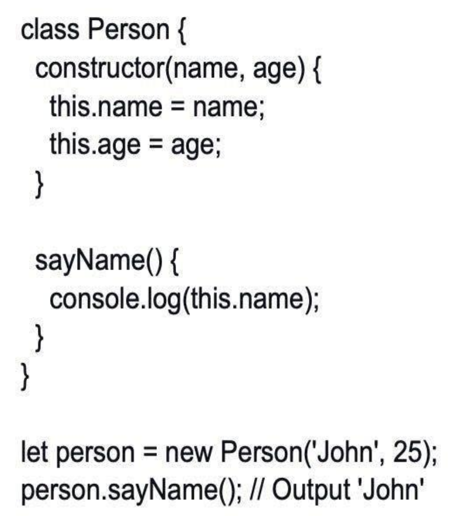
130. Explain how DOM is utilized in JavaScript.

Document Object Model (DOM) is how different objects in a document interact with each other. It is used for developing web pages that include objects like paragraphs, links, etc. These Objects can be made to perform actions like add or delete. Also, it adds extra abilities to a web page.

131. How would you define a class in JavaScript?

To define a class in JavaScript, you can use the class keyword. For example:

CYBROM



```
class Person {
  constructor(name, age) {
    this.name = name;
    this.age = age;
  }

  sayName() {
    console.log(this.name);
  }
}

let person = new Person('John', 25);
person.sayName(); // Output 'John'
```

132. Explain event bubbling.

In JavaScript, DOM elements can be nested inside each other.
Now, in such cases, when the handler of the child is clicked, the handler associated with the parent will also work.

133. Define window.onload and onDocumentReady.

Window.onload is an event that is fired when the page has finished loading. This event is often used to perform tasks that need to be done after the page has loaded, such as setting up event handlers, initializing components, or making AJAX requests.

On the contrary, onDocumentReady is an event that is fired when the page's HTML document is ready to be interacted with. This event is often used to perform tasks that need to be done when the page is ready, such as setting up event handlers, initializing components, or making AJAX requests.

134. How can you append a value to an array?

Hide Answer

We can append a value to an array by

arr[arr.length] = value;

135. List some features of JavaScript.

Some of the features of JavaScript are:

- Lightweight
- Interpreted programming language
- Good for the applications which are network-centric
- Complementary to Java
- Complementary to HTML
- Open source

Cross-platform

---

136. Who developed JavaScript, and what was the first name of JavaScript?

- JavaScript was developed by Brendan Eich, who was a Netscape programmer. Brendan Eich developed this new scripting language in just ten days in the year September 1995. At the time of its launch, JavaScript was initially called Mocha. After that, it was called Live Script and later known as JavaScript.

---

137. List some of the advantages of JavaScript.

Some of the advantages of JavaScript are:

- Server interaction is less
- Feedback to the visitors is immediate
- Interactivity is high
- Interfaces are richer

---

138. List some of the disadvantages of JavaScript.

- Some of the disadvantages of JavaScript are:
- No support for multithreading
- No support for multiprocessing
- Reading and writing of files is not allowed
- No support for networking applications.

139. Define a named function in JavaScript.

The function which has named at the time of definition is called a named function. For example

```
function msg()

{

  document.writeln("Named Function");

}

msg();
```

140. Name the types of functions

The types of function are:

Named - These type of functions contains name at the time of definition. For Example:

```
function display()
{
  document.writeln("Named Function");
}
display();
```

Anonymous - These type of functions doesn't contain any name. They are declared dynamically at runtime.

```
var display=function()
{
  document.writeln("Anonymous Function");
}
display();
```

141. Define anonymous function

It is a function that has no name. These functions are declared dynamically at runtime using the function operator instead of the function declaration. The function operator is more flexible than a function declaration. It can be easily used in the place of an expression. For example:

```
var display=function()
{
 alert("Anonymous Function is invoked");
}
display();
```

142. Can an anonymous function be assigned to a variable?

Yes, you can assign an anonymous function to a variable.

143. Define closure.

In JavaScript, we need closures when a variable which is defined outside the scope in reference is accessed from some inner scope.

```
var num = 10;
function sum()
{
document.writeln(num+num);
}
sum();
```

144. What are the key differences between Java and JavaScript? / How is JavaScript different from Java?

JavaScript is a lightweight programming language (most commonly known as scripting language) developed by Netscape, Inc. It is used to make web pages interactive. It is not a part of the Java platform. Following is a list of some key differences between Java and JavaScript

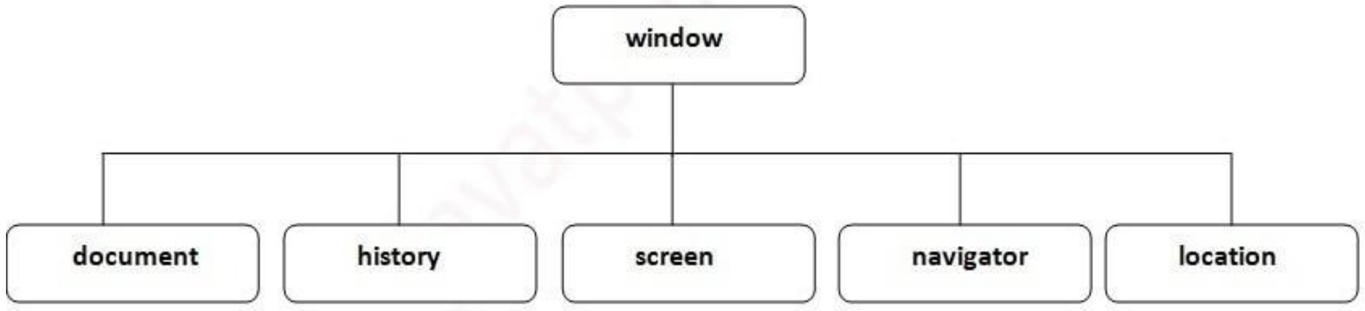
A list of key differences between Java and JavaScript

Java	JavaScript
Java is a complete and strongly typed programming language used for backend coding. In Java, variables must be declared first to use in the program, and the type of a variable is checked at compile-time.	JavaScript is a weakly typed, lightweight programming language (most commonly known as scripting language) and has more relaxed syntax and rules.
Java is an object-oriented programming (OOPS) language or structured programming languages such as C, C++, or .Net.	JavaScript is a client-side scripting language, and it doesn't fully support the OOPS concept. It resides inside the HTML documents and is used to make web pages interactive (not achievable with simple HTML).
Java creates applications that can run in any virtual machine (JVM) or browser.	JavaScript code can run only in the browser, but it can now run on the server via Node.js.
The Java code needs to be compiled.	The JavaScript code doesn't require to be complied.
Java Objects are class-based. You can't make any program in Java without creating a class.	JavaScript Objects are prototype-based.
Java is a Complete and Standalone language that can be used in backend coding.	JavaScript is assigned within a web page and integrates with its HTML content.
Java programs consume more memory.	JavaScript code is used in HTML web pages and requires less memory.
The file extension of the Java program is written as ".Java" and it translates source code into bytecodes which are then executed by JVM (Java Virtual Machine).	The JavaScript file extension is written as ".js" and it is interpreted but not compiled. Every browser has a JavaScript interpreter to execute the JS code.
Java supports multithreading.	JavaScript doesn't support multithreading.
Java uses a thread-based approach to concurrency.	JavaScript uses an event-based approach to concurrency.

145. What is BOM?

BOM stands for Browser Object Model. It provides interaction with the browser. The default object of a browser is a window. So, you can call all the functions of the window by specifying the window or directly. The window object provides various properties like document, history, screen, navigator, location, innerHeight, innerWidth,

Advertisement



146. What is DOM? What is the use of document object?

DOM stands for Document Object Model. A document object represents the HTML document. It can be used to access and change the content of HTML.

147. What is the use of window object?

The window object is created automatically by the browser that represents a window of a browser. It is not an object of JavaScript. It is a browser object.

The window object is used to display the popup dialog box. Let's see with description.

Method	Description
alert()	displays the alert box containing the message with ok

	button.
confirm()	displays the confirm dialog box containing the message with ok and cancel button.
prompt()	displays a dialog box to get input from the user.
open()	opens the new window.
close()	closes the current window.
setTimeout()	performs the action after specified time like calling function, evaluating expressions.

More details.

---

148. What is the use of history object?

The history object of a browser can be used to switch to history pages such as back and forward from the current page or another page. There are three methods of history object.

1. history.back() - It loads the previous page.
2. history.forward() - It loads the next page.
3. history.go(number) - The number may be positive for forward, negative for backward. It loads the given page number.

---

149. How to write HTML code dynamically using JavaScript?

The innerHTML property is used to write the HTML code using JavaScript dynamically. Let's see a simple example:

document.getElementById('mylocation').innerHTML="<h2>This is heading using J
avaScript</h2>";

---

150. How to create objects in JavaScript?

There are 3 ways to create an object in JavaScript.

1. By object literal
2. By creating an instance of Object
3. By Object Constructor

Let's see a simple code to create an object using object literal.

```
emp={id:102,name:"Rahul Kumar",salary:50000}
```

---

151. What does the isNaN() function?

The isNan() function returns true if the variable value is not a number. For example:

```
function number(num) {
  if (isNaN(num)) {
    return "Not a Number";
  }
  return "Number";
}
console.log(number('1000F'));
// expected output: "Not a Number"

console.log(number('1000'));
// expected output: "Number"
```

---

152. What is the output of 10+20+"30" in JavaScript?

3030 because 10+20 will be 30. If there is numeric value before and after +, it treats as binary + (arithmetic operator).

```
function display()
{
  document.writeln(10+20+"30");
}
```

```
display();
```

---

153. What is the output of "10"+20+30 in JavaScript?

102030 because after a string all the + will be treated as string concatenation operator (not binary +).

```
function display()
{
  document.writeln("10"+20+30);
}
display();
```

---

154. Difference between Client side JavaScript and Server side JavaScript?

Client-side JavaScript comprises the basic language and predefined objects which are relevant to running JavaScript in a browser. The client-side JavaScript is embedded directly by in the HTML pages. The browser interprets this script at runtime.
Server-side JavaScript also resembles client-side JavaScript. It has a relevant JavaScript which is to run in a server. The server-side JavaScript are deployed only after compilation.

---

155. What's the difference between event.preventDefault() and event.stopPropagation() methods in JavaScript?

In JavaScript, the event.preventDefault() method is used to prevent the default behavior of an element.

For example: If you use it in a form element, it prevents it from submitting. If used in an anchor element, it prevents it from navigating. If used in a contextmenu, it prevents it from showing or displaying.

On the other hand, the event.stopPropagation() method is used to stop the propagation of an event or stop the event from occurring in the bubbling or capturing phase.

---

156. What is the real name of JavaScript?

The original name was Mocha, a name chosen by Marc Andreessen, founder of Netscape. In September of 1995, the name was changed to LiveScript. In December 1995, after receiving a trademark license from Sun, the name JavaScript was adopted.

157. What is the difference between View state and Session state?

"View state" is specific to a page in a session whereas "Session state" is specific to a user or browser that can be accessed across all pages in the web application.

158. How to submit a form using JavaScript by clicking a link?

Let's see the JavaScript code to submit the form by clicking the link.

```
<form name="myform" action="index.php">
Search: <input type='text' name='query' />
<a href="javascript: submitform()">Search</a>
</form>
<script type="text/javascript">
function submitform()
{
  document.myform.submit();
}
</script>
```

---

159. Is JavaScript faster than ASP script?

Yes, because it doesn't require web server's support for execution.

---

160. How to change the background color of HTML document using JavaScript?

```
<script type="text/javascript">
```

```
document.body.bgColor="pink";

</script>
```

---

161. How to handle exceptions in JavaScript?

By the help of try/catch block, we can handle exceptions in JavaScript. JavaScript supports try, catch, finally and throw keywords for exception handling.

---

162. How to validate a form in JavaScript?

```
<script>
function validateform(){
var name=document.myform.name.value;
var password=document.myform.password.value;

if (name==null || name==""){
  alert("Name can't be blank");
  return false;
}else if(password.length<6){
  alert("Password must be at least 6 characters long.");
  return false;
  }
}
</script>
<body>
<form name="myform" method="post" action="abc.jsp" onsubmit="return valida
teform()" >
Name: <input type="text" name="name"><br/>
Password: <input type="password" name="password"><br/>
<input type="submit" value="register">
</form>
```

---

163. How to validate email in JavaScript?

CYBROM

```
<script>
function validateemail()
{
var x=document.myform.email.value;
var atposition=x.indexOf("@");
var dotposition=x.lastIndexOf(".");
if (atposition<1 || dotposition<atposition+2 || dotposition+2>=x.length){
  alert("Please enter a valid e-
mail address \n atpostion:"+atposition+"\n dotposition:"+dotposition);
  return false;
  }
}
</script>
<body>
<form name="myform"  method="post" action="#" onsubmit="return validateem
ail();">
Email: <input type="text" name="email"><br/>

<input type="submit" value="register">
</form>
```

---

164. What is the requirement of debugging in JavaScript?

JavaScript didn't show any error message in a browser. However, these mistakes can affect the output. The best practice to find out the error is to debug the code. The code can be debugged easily by using web browsers like Google Chrome, Mozilla Firebox.

To perform debugging, we can use any of the following approaches:

- Using console.log() method
- Using debugger keyword

---

165. What is the use of debugger keyword in JavaScript?

CYBROM

JavaScript debugger keyword sets the breakpoint through the code itself. The debugger stops the execution of the program at the position it is applied. Now, we can start the flow of execution manually. If an exception occurs, the execution will stop again on that particular line.. For example:

```
function display()
{
x = 10;
y = 15;
z = x + y;
debugger;
document.write(z);
document.write(a);
}
display();
```

---

166. What is the role of a strict mode in JavaScript?

The JavaScript strict mode is used to generates silent errors. It provides "use strict"; expression to enable the strict mode. This expression can only be placed as the first statement in a script or a function. For example:

```
"use strict";
x=10;
console.log(x);
```

---

167. What is the use of Math object in JavaScript?

The JavaScript math object provides several constants and methods to perform a mathematical operation. Unlike date object, it doesn't have constructors. For example:

```
function display()
{
 document.writeln(Math.random());
}
```

CYBROM

```
display();
```

168.What is the use of a Date object in JavaScript?

The JavaScript date object can be used to get a year, month and day. You can display a timer on the webpage by the help of JavaScript date object.

```
function display()
{
 var date=new Date();
var day=date.getDate();
var month=date.getMonth()+1;
var year=date.getFullYear();
document.write("<br>Date is: "+day+"/"+month+"/"+year);
}
display();
```

---

169. What is the use of a Number object in JavaScript?

The JavaScript number object enables you to represent a numeric value. It may be integer or floating-point. JavaScript number object follows the IEEE standard to represent the floating-point numbers.

```
function display()
{
var x=102;//integer value
var y=102.7;//floating point value
var z=13e4;//exponent value, output: 130000
var n=new Number(16);//integer value by number object
document.write(x+" "+y+" "+z+" "+n);
}
display();
```

---

170 .What is the use of a Boolean object in JavaScript?

CYBROM

The JavaScript Boolean is an object that represents value in two states: true or false. You can create the JavaScript Boolean object by Boolean() constructor.

```
function display()
{
document.writeln(10<20);//true
document.writeln(10<5);//false
}
display();
```

---

171. What is the use of a TypedArray object in JavaScript?

The JavaScript TypedArray object illustrates an array like a view of an underlying binary data buffer. There is any number of different global properties, whose values are TypedArray constructors for specific element types.

```
function display()
{
var arr1= [1,2,3,4,5,6,7,8,9,10];
    arr1.copyWithin(2) ;
    document.write(arr1);
}
display();
```

---

172. What is the use of a Set object in JavaScript?

The JavaScript Set object is used to store the elements with unique values. The values can be of any type i.e. whether primitive values or object references. For example:

```
function display()
{
var set = new Set();
set.add("jQuery");
set.add("AngularJS");
set.add("Bootstrap");
for (let elements of set) {
 document.writeln(elements+"<br>");
}
}
display();
```

---

173. What is the use of a WeakSet object in JavaScript?

The JavaScript WeakSet object is the type of collection that allows us to store weakly held objects. Unlike Set, the WeakSet are the collections of objects only. It doesn't contain the arbitrary values. For example:

```
function display()
{
var ws = new WeakSet();
var obj1={};
var obj2={};
ws.add(obj1);
ws.add(obj2);
//Let's check whether the WeakSet object contains the added object
document.writeln(ws.has(obj1)+"<br>");
document.writeln(ws.has(obj2));
}
display()
```

---

174. What is the use of a WeakMap object in JavaScript?

The JavaScript WeakMap object is a type of collection which is almost similar to Map. It stores each element as a key-value pair where keys are weakly referenced. Here, the keys are objects and the values are arbitrary values. For example:

```
function display()
{
var wm = new WeakMap();
var obj1 = {};
var obj2 = {};
var obj3= {};
```

CVBROM

```
wm.set(obj1, "jQuery");
wm.set(obj2, "AngularJS");
wm.set(obj3,"Bootstrap");
document.writeln(wm.has(obj2));
}
display();
```

---

175. What are the falsy values in JavaScript, and how can we check if a value is falsy?

Those values which become false while converting to Boolean are called falsy values.

const falsyValues = ['', 0, null, undefined, NaN, false];

We can check if a value is falsy by using the Boolean function or the Double NOT operator (!!).

---

176. What do you understand by hoisting in JavaScript?

Hoisting is the default behavior of JavaScript where all the variable and function declarations are moved on top. In simple words, we can say that Hoisting is a process in which, irrespective of where the variables and functions are declared, they are moved on top of the scope. The scope can be both local and global.

Example 1:

```
hoistedVariable = 12;
console.log(hoistedVariable); // outputs 12 even when the variable is declared aft
er it is initialized
var hoistedVariable;
```

Example2:

```
hoistedFunction();  // Outputs " Welcome to JavaTpoint " even when the function
is declared after calling
function hoistedFunction(){
  console.log(" Welcome to JavaTpoint ");
}
```

Example3:

```
// Hoisting in a local scope
function doSomething(){
```

CYBROM

```
  x = 11;
  console.log(x);
  var x;
}
doSomething(); // Outputs 11 since the local variable "x" is hoisted inside the local
 scope
```

177. What are some of the built-in methods in JavaScript?

Built-in Method	Values
Date()	Returns the present date and time
concat()	Joins two strings and returns the new string
push()	Adds an item to an array
pop()	Removes and also returns the last element of an array
round()	Rounds of the value to the nearest integer and

CYBROM

	then returns it
length()	Returns the length of a string

178. What are the scopes of a variable in JavaScript?

The scope of a variable implies where the variable has been declared or defined in a JavaScript program. There are two scopes of a variable:

Global Scope

Global variables, having global scope are available everywhere in a JavaScript code.

Local Scope

Local variables are accessible only within a function in which they are defined.

179. What are the conventions of naming a variable in JavaScript?

Following are the naming conventions for a variable in JavaScript:

- Variable names cannot be similar to that of reserved keywords. For example, var, let, const, etc.
- Variable names cannot begin with a numeric value. They must only begin with a letter or an underscore character.
- Variable names are case-sensitive.

.

180. What are some of the JavaScript frameworks and their uses?

CYBROM

JavaScript has a collection of many frameworks that aim towards fulfilling the different aspects of the web application development process. Some of the prominent frameworks are:

- React - Frontend development of a web application
- Angular - Frontend development of a web application
- Node - Backend or server-side development of a web application

181. What is the difference between Call and Apply? (explain in detail with examples)

- Call

Call uses arguments separately.

Example:

```
function sayHello()

{

  return "Hello " + this.name;

}

var obj = {name: "Sandy"};

sayHello.call(obj);

// Returns "Hello Sandy"
```

- Apply

Apply uses an argument as an array.

Example:

```
function saySomething(message)
```

CYBROM

```
{
  return this.name + " is " + message;
}
var person4 = {name:  "John"};
saySomething.apply(person4, ["awesome"]);
```

182. What are the scopes of a variable in JavaScript?

The scope of variables in JavaScript is used to determine the accessibility of variables and functions at various parts of one’s code. There are three types of scopes of a variable, global scope, function scope, block scope

- Global Scope: It is used to access the variables and functions from anywhere inside the code.

Example:

```
var globalVariable = "Hello world";
function sendMessage(){
  return globalVariable; // globalVariable is accessible as it's written in global space
}
function sendMessage2(){
  return sendMessage(); // sendMessage function is accessible as it's written in global space
}
sendMessage2();  // Returns “Hello world”
```

CYBROM

- Function scope: It is used to declare the function and variables inside the function itself and not outside.

Example:

```
function awesomeFunction()

{

  var a = 3;

  var multiplyBy3 = function()

{

    console.log(a*3); // Can access variable "a" since a and multiplyBy3 both are written inside the same function

  }

}

console.log(a); // a is written in local scope and can't be accessed outside and throws a reference error

multiplyBy3(); // MultiplyBy3 is written in local scope thus it throws a reference error
```

- Block Scope: It uses let and const to declare the variables.

Example:

```
{

  let x = 45;

}

console.log(x); // Gives reference error since x cannot be accessed outside of the block
```

CYBROM

```
for(let i=0; i<2; i++){

  // do something

}

console.log(i); // Gives reference error since i cannot be accessed outside of the
for loop block
```

- keyword has been used from the very initial stages of JavaScript.
- We can perform functions with the help of the keyword "var' by accessing various variables.
- Keyword "let"
- The Keyword "let" was added later in ECMAScript 2015 in JavaScript Programming.
- Variable declaration is very limited with the help of the "let" keyword that is declared in Block. Also, it might result in a ReferenceError as the variable was declared in the "temporal dead zone" at the beginning of the block.

183. Passed by value and passed by reference

- Passed By Values Are Primitive Data Types.

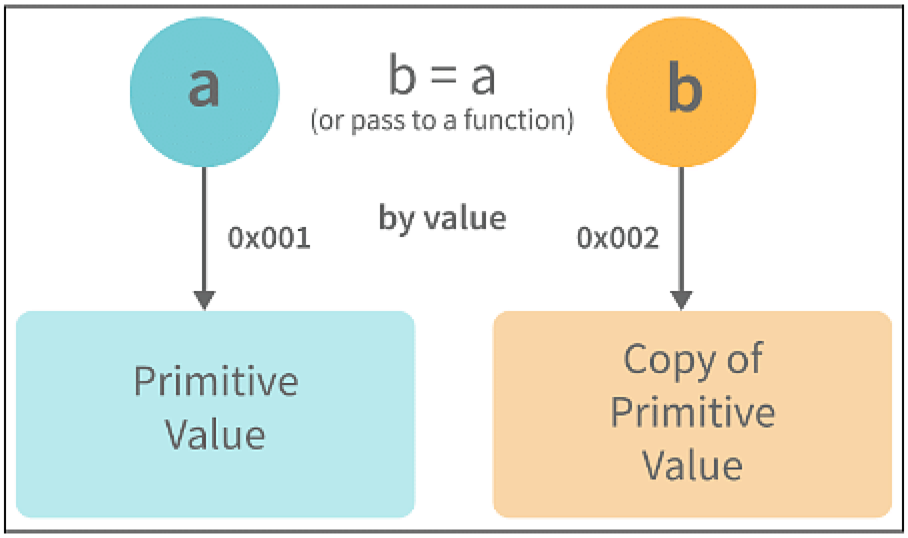
Consider the following example:

Here, the a=432 is a primitive data type i.e. a number type that has an assigned value by the operator. When the var b=a code gets executed, the value of 'var a' returns a new address for 'var b' by allocating a new space in the memory, so that 'var b' will be operated at a new location.

Example:

```
var a = 432;

var b = a;
```

CYBROM



- Passed by References Are Non-primitive Data Types.

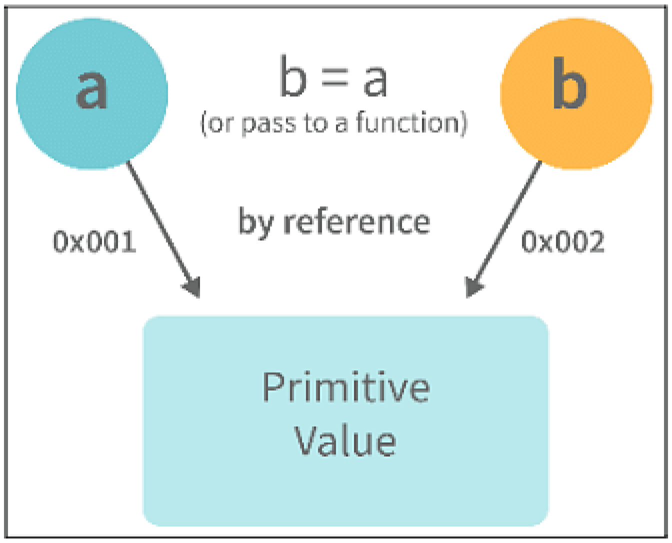
Consider the following example:

The reference of the 1st variable object i.e. 'var obj' is passed through the location of another variable i.e. 'var obj2' with the help of an assigned operator.

Example:

```
var obj = { name: "Raj", surname: "Sharma" };

var obj2 = obj;
```

CYBROM



184 Characteristics of javascript strict-mode

Strict mode does not allow duplicate arguments and global variables.

- One cannot use JavaScript keywords as a parameter or function name in strict mode.
- All browsers support strict mode.
- Strict mode can be defined at the start of the script with the help of the keyword ‘use strict’.

185. Higher Order Functions (with examples)

- Higher-order functions are the functions that take functions as arguments and return them by operating on other functions
  Example:
  ```
  function higherOrder(fn)
   {
    fn();
  }
  higherOrder(function() { console.log("Hello world") });
  ```

186. Self Invoking Functions

CYBROM

Self Invoking Functions is an automatically invoked function expression followed by (), where it does not need to be requested. Nevertheless, the declaration of the function is not able to be invoked by itself.

187. difference between exec () and test () methods

- exec()
- It is an expression method in JavaScript that is used to search a string with a specific pattern.
- Once it has been found, the pattern will be returned directly, otherwise, it returns an “empty” result.
- test ()
- It is an expression method in JavaScript that is also used to search a string with a specific pattern or text.
- Once it has been found, the pattern will return the Boolean value 'true', else it returns ‘false’.

188. What is memoization?

In JavaScript, when we want to cache the return value of a function concerning its parameters, it is called memoization. It is used to speed up the application especially in case of complex, time consuming functions.

189. Use of a constructor function (with examples)

Constructor functions are used to create single objects or multiple objects with similar properties and methods.

Example:

```
function Person(name,age,gender)

{
```

CYBROM

```
  this.name = name;

  this.age = age;

  this.gender = gender;

}

var person1 = new Person("Vivek", 76, "male");

console.log(person1);

var person2 = new Person("Courtney", 34, "female");

console.log(person2);
```

190. Difference between client-side and server-side

- Client-side JavaScript
- Client-side JavaScript is made up of fundamental language and predefined objects that perform JavaScript in a browser.
- Also, it is automatically included in the HTML pages where the browser understands the script.
- Server-side Javascript
- Server-side JavaScript is quite similar to Client-side javascript.
- Server-side JavaScript can be executed on a server.
- The server-side JavaScript is deployed once the server processing is done.

191. Rest parameter and spread operator

- Rest Parameter(...)
- Rest parameter is used to declare the function with improved handling of parameters.

CYBROM

- Rest parameter syntax can be used to create functions to perform functions on the variable number of arguments.
- It also helps to convert any number of arguments into an array as well as helps in extracting some or all parts of the arguments.
- Spread Operator(...)
- In a function call, we use the spread operator.
- It's also to spread one or more arguments that are expected in a function call.
- The spread operator is used to take an array or an object and spread them.

192. Promises in JavaScript

- Promises in JavaScript have four different states. They are as follows:

• Pending	• Fulfilled	• Rejected	• Settled
• Pending is an initial state of promise. It is the initial state of promise where it is in the pending state that neither is fulfilled nor rejected.	• • It is the state where the promise has been fulfilled that assures that the async operation is done.	• It is the state where the promise is rejected and the async operation has failed.	• It is the state where the promis e is rejecte d or fulfilled .

- Example:
- function sumOfThreeElements(...elements)

CYBROM

- {
-   return new Promise((resolve,reject)=>{
-     if(elements.length > 3)
- {
-       reject("Just 3 elements or less are allowed");
-     }
-     else
- {
-       let sum = 0;
-       let i = 0;
-       while(i < elements.length)
- {
-         sum += elements[i];
-         i++;
-       }
-       resolve("Sum has been calculated: "+sum);
-     }
-   })
- }

193. What is Object Destructuring? (with examples)

Object destructuring is a method to extract elements from an array or an object.

Example 1: Array Destructuring

const arr = [1, 2, 3];

const first = arr[0];

CYBROM

const second = arr[1];

const third = arr[2];

Example 2: Object Destructuring

const arr = [1, 2, 3];

const [first,second,third,fourth] = arr;

console.log(first); // Outputs 1

console.log(second); // Outputs 2

console.log(third); // Outputs 3

194. What is a Temporal Dead Zone?

Temporal Dead Zone is a behavior that occurs with variables declared using let and const keywords before they are initialized.

195. What is Lexical Scoping?

Lexical Scoping in JavaScript can be performed when the internal state of the JavaScript function object consists of the function's code as well as references concerning the current scope chain.

196. What is this [[[]]]?

This '[[[]]]' is a three-dimensional array.

197. How do you empty an array in JavaScript?

There are a few ways in which we can empty an array in JavaScript:

- By assigning array length to 0:

var arr = [1, 2, 3, 4];

CYBROM

```
arr.length = 0;
```

- By assigning an empty array:

```
var arr = [1, 2, 3, 4];

arr = [];
```

- By popping the elements of the array:

```
var arr = [1, 2, 3, 4];

while (arr.length > 0) {

  arr.pop();

}
```

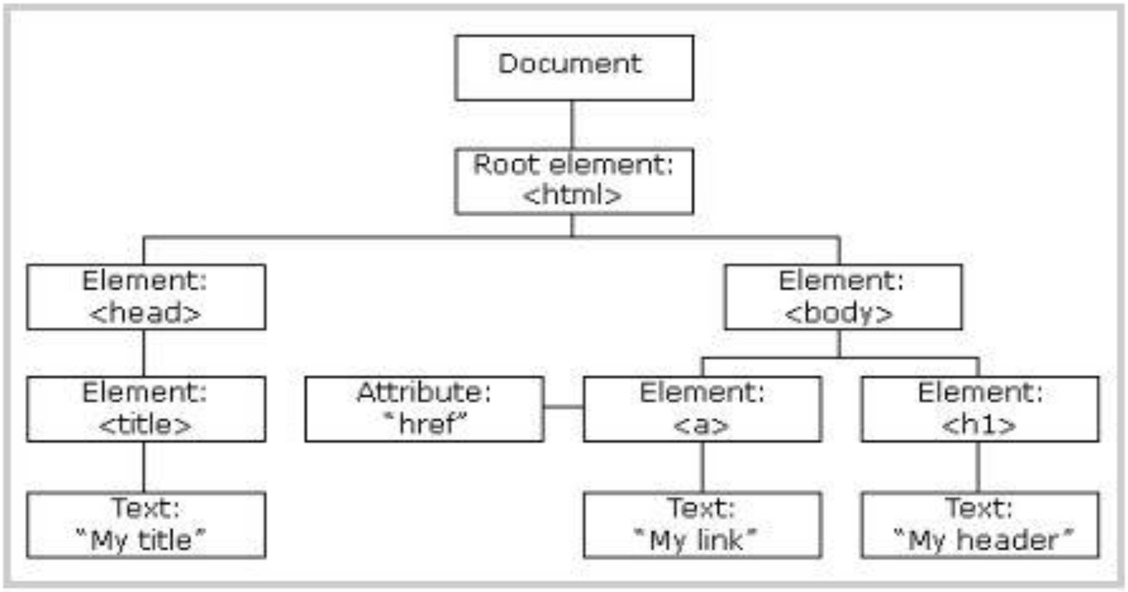
- By using the splice array function:

```
var arr = [1, 2, 3, 4];

arr.splice(0, arr.length);
```

198. What is the difference between Event Capturing and Event Bubbling?

Event Capturing	Event Bubbling
This process starts with capturing the event of the outermost element and then propagating it to the innermost element.	This process starts with capturing the event of the innermost element and then propagating it to the outermost element.

199. Can you draw a simple JavaScript DOM (Document Object Model)?



CYBRO