Beginner Level

1. **What is JavaScript?**

   JavaScript is a high-level, interpreted programming language primarily used for creating interactive web pages.

2. **What are the different data types in JavaScript?**

   JavaScript supports various data types including strings, numbers, booleans, null, undefined, objects, and symbols (added in ES6).

3. **How do you declare variables in JavaScript?**

   Variables in JavaScript can be declared using the `var`, `let`, or `cons` keywords.

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

   `var` has function scope, `let` has block scope, and `const` is used to declare constants whose value cannot be reassigned.

5. **What is the use of `==` and `===` operators in JavaScript?**

   `==` is used for loose equality comparison, whereas `===` is used for strict equality comparison, which checks both value and type.

6. **How do you comment in JavaScript?**

   Single-line comments are denoted by `//`, and multi-line comments are enclosed between `/*` and `*/`.

7. **What are JavaScript primitive data types?**

   Primitive data types in JavaScript include strings, numbers, booleans, null, undefined, and symbols.

8. **What is the `typeof` operator used for?**

   The `typeof` operator is used to determine the type of a variable or an expression.

9. **Explain hoisting in JavaScript.**

Hoisting is JavaScript's default behavior of moving variable and function declarations to the top of their containing scope during compilation.

10. **What is NaN in JavaScript?**

    `NaN` stands for "Not-a-Number" and is a value returned when a mathematical operation cannot produce a meaningful result.

Intermediate Level

11. **Explain closures in JavaScript.**

    Closures are functions that have access to their outer function's scope, even after the outer function has finished executing.

12. **What are higher-order functions in JavaScript?**

    Higher-order functions are functions that operate on other functions, either by taking them as arguments or by returning them.

13. **What is a callback function?**

    A callback function is a function passed as an argument to another function, which is then invoked inside the outer function to complete some kind of routine or action.

14. **What is event delegation in JavaScript?**

    Event delegation is a technique where a single event listener is attached to a parent element to listen for events that occur on its children. This is particularly useful when dealing with dynamically added elements.

15. **Explain the concept of prototypal inheritance in JavaScript.**

    Prototypal inheritance is the mechanism by which objects in JavaScript inherit properties and methods from other objects, known as prototypes.

16. **What is the `this` keyword in JavaScript?**

    The `this` keyword refers to the object on which a method is being invoked or the context in which a function is called.

17. **How does JavaScript handle asynchronous operations?**

JavaScript uses mechanisms such as callbacks, promises, and async/await to handle asynchronous operations.

18. **What are JavaScript promises?**

    Promises are objects representing the eventual completion or failure of an asynchronous operation, and they allow for more readable and manageable asynchronous code.

19. **Explain the difference between `null` and `undefined` in JavaScript.**

    `null` represents the intentional absence of any object value, while `undefined` represents the absence of a defined value.

20. **What is the event loop in JavaScript?**

    The event loop is a mechanism that handles asynchronous operations in JavaScript, ensuring that they are executed in a non-blocking manner.

21. What is the difference between `null`, `undefined`, and `undeclared` in JavaScript?

`null` is an explicitly assigned value that represents the absence of any object value, `undefined` indicates a variable that has been declared but has not yet been assigned a value, and `undeclared` refers to variables that have not been declared at all.

22. Explain the concept of event-driven programming in JavaScript.

Event-driven programming is a paradigm where the flow of the program is determined by events such as user actions, network requests, or timer events, rather than being strictly sequential.

23. What is the difference between synchronous and asynchronous JavaScript?

Synchronous JavaScript executes code sequentially, blocking further execution until the current operation is completed, whereas asynchronous JavaScript allows multiple operations to be executed concurrently, without blocking the main execution thread.

24. How do you handle errors in JavaScript?

Errors in JavaScript can be handled using try-catch blocks to catch exceptions and handle them gracefully, preventing the program from crashing.

25. What are JavaScript modules and how do they improve code organization?

JavaScript modules are reusable pieces of code that encapsulate related functionality and are designed to promote modularity, encapsulation, and code reuse, leading to better code organization and maintainability

Advanced Level

26. What are generators in JavaScript?

Generators are functions that can be paused and resumed, allowing for more flexible control flow.

27. What are arrow functions in JavaScript?

Arrow functions are a concise way to write anonymous functions in JavaScript, introduced in ES6.

28. Explain the concept of currying in JavaScript.

Currying is the technique of converting a function that takes multiple arguments into a sequence of functions that each take a single argument.

29. What is memoization and how is it useful in JavaScript?

Memoization is an optimization technique used to speed up function execution by caching the results of expensive function calls and returning the cached result when the same inputs occur again.

30. What are closures and how are they implemented in JavaScript?

Closures are functions that have access to the outer function's variables even after the outer function has finished executing. They are implemented by creating a scope chain that remains intact even after the outer function exits.

31. What are modules in JavaScript?

Modules are reusable pieces of code that encapsulate related functionality. They help in organizing code, managing dependencies, and promoting reusability.

32. Explain the difference between `null` and `undefined` in JavaScript.

`null` is an explicitly assigned value that represents the absence of any object value, while `undefined` indicates a variable that has been declared but has not yet been assigned a value.

33. What are the different ways to create objects in JavaScript?

Objects in JavaScript can be created using object literals, constructor functions, the `Object.create()` method, and class syntax (introduced in ES6).

34. What is event bubbling and event capturing in JavaScript?

Event bubbling is a mechanism where an event occurring in a child element is propagated up to its parent elements, while event capturing is the reverse process where the event is captured from the top of the DOM hierarchy down to the target element.

35. Explain the concept of `bind`, `call`, and `apply` methods in JavaScript.

`bind`, `call`, and `apply` are methods used to manipulate the `this` keyword in JavaScript functions. `bind` creates a new function with a specified `this` value, `call` calls a function with a specified `this` value and arguments passed individually, and `apply` calls a function with a specified `this` value and arguments passed as an array.

36. Explain the concept of function currying with an example.

Function currying is the process of transforming a function that takes multiple arguments into a series of functions that each take a single argument. For example:

```
function multiply(a) {
        return function(b) {
        return a * b;
        };
}

const multiplyByTwo = multiply(2);
console.log(multiplyByTwo(3)); // Output: 6
```

37. What are closures and how can they lead to memory leaks in JavaScript?

Closures are functions that have access to their outer function's scope even after the outer function has finished executing. They can lead to memory leaks if they hold references to large objects or variables that are no longer needed, preventing them from being garbage collected.

38. Explain the concept of prototypal inheritance and how it differs from classical inheritance.

Prototypal inheritance is a mechanism in JavaScript where objects inherit properties and methods from other objects through prototype chains. It differs

from classical inheritance, which involves the creation of classes and instances, by being more flexible and dynamic.

39. What are the differences between `==` and `===` in JavaScript? Provide examples.

`==` performs type coercion before comparison, while` ===` does not. For example:

```
console.log(1 == '1');  // Output: true (coerces string to number)
console.log(1 === '1'); // Output: false (strict comparison)
```

40. Explain the difference between the `for...in` and `for...of` loops in JavaScript.

The `for...in` loop iterates over the enumerable properties of an object, while the `for...of` loop iterates over iterable objects such as arrays, strings, maps, and sets, returning their property values.