**Prerequisities**

**Questions on HTML**

**Questions on CSS**

**Questions on JAVASCRIPT**

**1 What is Javascript and its purpose?**

Programming language primarily used to create interactive and dynamic content on UI.

**Purpose of JavaScript:**

* **Client-Side Scripting**: JavaScript is mainly used for client-side development, it runs in the user's web browser without the need for server-side processing.
* **DOM Manipulation**: JavaScript can interact with the Document Object Model (DOM), enabling developers to modify the content and structure of web pages dynamically.
* **Event Handling**: JavaScript allows developers to respond to user interactions such as clicks, key presses, or form submissions, making web pages more interactive.
* **Form Validation**: JavaScript can validate user input in forms before submitting data to the server
* **Asynchronous Operations**: JavaScript supports asynchronous programming through techniques like callbacks, promises, and async/await.
* **Web APIs Integration**: JavaScript can interact with various web APIs (e.g., Geolocation API, Fetch API, Web Storage API) to enhance functionality, such as retrieving data from a server, storing data locally, or accessing the user’s location.
* **Cross-Platform Development**: With environments like Node.js, JavaScript can also be used for server-side development, enabling the creation of full-stack applications. It's also used in mobile app development (e.g., React Native) and desktop applications (e.g., Electron).

**2. Explain what kind of application we can built using Javascript.**

variety of applications across different platforms.

* Web Application
* Mobile Application
* Desktop Applications etc

**3. What are variables and explain the rules to create a variable**

Variables in JavaScript are used to store data that can be referenced and manipulated within a program.

Rules to follow while making use of variables

* Variable name should contain only letter,digits,symbols($) and (\_)underscore
* First character must not be the digits
* Variable Names should be in camelCase

**4. What are Data types**

Data types define the kind of data that can be stored and manipulated within a program.

* Primitive Data Types:
* Non-Primitive Data Types

**5. Explain Primitive and Non Primitive Data type.**

Data type used to store single value are known as Primitive data types Primitive Data Types: String, Number, Boolean, Undefined, Null, Symbol, BigInt.

Data types are used to store multiple values are known as Non Primitive Data types

Non-Primitive Data Types: Object (which includes Arrays, Functions, Dates, etc.).

**6. What is function in Javascript**

Reusable block of code designed to perform a specific task. Functions allow you to encapsulate logic and avoid code repetition

**7. Explain the Difference between callback and higherorder function in JS?**

Callback Functions

A callback function is a function that is passed as an argument to another function.

Callbacks are often used for asynchronous operations, like handling events, making API requests, or performing tasks after a delay.

callback function is executed after some operation has been completed.

A **higher-order function** is a function that either:

1. Takes one or more functions as arguments, or
2. Returns a function as its result.

**8. What is difference between let ,cons and var**

* Hoisting
* Scope/Memory
* Redeclaration/ Updation

Hoisting means it is a mechanism where it moves the variable/function declaration to the top of its nearest scope

Example

|  |  |  |
| --- | --- | --- |
| **Var** | **Let** | **Const** |
| console.log(a); | HOISTING will not support | |
| var a = 10; |
| Internally var a; console.log(a);//nearest scope a = 10 ; |
| SCOPE/Memory var is a function level scope | let/const are block level scope { } | |
| var supports redeclaration and updation | let supports only updation | with const no redeclaration / updation |

**9. What is spread operator**

Getting new array with the existing array values but not same array.

the spread operator is used to expand elements from an array or object into individual elements.

**10. What is Rest Parameter**

Rest parameter is used in function definitions to gather multiple arguments into an array, whereas

**11. What is promise in Javascript**

Promises simplify the handling of asynchronous code. if code tooks long time then the below code needs to wait for the current execution.

 Methods:

* then(): Handles fulfillment and optionally rejection.
* catch(): Handles rejection.
* finally(): Runs regardless of fulfillment or rejection.

 Advanced Methods: Promise.all(), Promise.race(), Promise.allSettled(), Promise.any().

**12. Explain different States of Promise in Javascript**

• States: Pending, Fulfilled, Rejected.

**13. How to create a Promise**

const promise = new Promise((resolve, reject) => {

  setTimeout(() => {

    const input = "ABC";

    if (input) {

      resolve(input);

    } else {

      reject(new Error("no data"));

    }

  }, 5000);

});

promise

  .then((input) => {

    console.log(input);

  })

  .catch((err) => {

    console.log(err);

  })

  .finally(() => {

    console.log(Test);

  });

**14. What is difference between promise and callback in Javascript**

* Callbacks: Functions passed to other functions to be executed later. Can lead to deeply nested code and complex error handling.
* Promises: Objects that represent the completion of asynchronous operations. Provide better readability, chaining, and error handling.

Promises are generally preferred over callbacks for handling asynchronous operations due to their cleaner syntax, better error management, and improved composability.

**15. What is Callback**

A callback is a function that is passed as an argument to another function to be "called back" at a later time(exception,deadlock,successful)

**16. What is higher order function in Javascript**

* Higher-Order Function: A function that either accepts functions as arguments or returns a function.
  + Accepting Functions: Allows customization and flexibility by passing functions as arguments.
  + Returning Functions: Enables dynamic creation of functions with specialized behavior.

Higher-order functions enhance code modularity, readability, and reusability, making them an essential concept in modern JavaScript programming.

**17. Explain different types of function in Javascript**

functions are the building blocks

Allow not to repeat the code

Function Declaration and Function Expression

=> with function declaration calling is possible without declaration

=> with function expression calling is not possible without declaring it

**18. What is arrow function in Javascript**

Arrow functions are useful for creating short and readable functions, A concise way to write function expressions

Syntax: (parameters) => { body }

**19. Why we use call, apply bind method in**

call: Invokes the function immediately with a specified this and arguments.

apply: Invokes the function immediately with a specified this and an array of arguments.

bind: Creates a new function with a specified this and optional pre-filled arguments, to be invoked later.

**20. Javascript**

**21. How many way to create object in Javascript**

* Object Literal Notation: {} syntax.
* Using new Object(): new Object().
* Object.create(): Creates objects with a specified prototype.
* Constructor Functions: Functions used with the new keyword.
* ES6 Classes: class syntax.
* Factory Functions: Functions that return new objects.

**22. What are the array method , string method**

**23. What is difference between java and javascript**

**24. What is Null and undefined in javascript**

**25. What is setTimeOut and setInterval in Javascript**

**26. What is sessionStorage, localStorage**

**27. What are is map, filter , reducer in javascript**

**28. What is closure in Javascript**

**29. What is housing in Javascript**

**30. What is the difference between array and object**

**31. What is the difference between number type and BigInt type**

**32. What is the difference between call() and apply()**

**33. What is DOM and DOM Manipulation**

**34. How to create element dynamically using Javascript**

**35. Explain Closures in Javascript**

**36. Explain different Scopes in Javascript**

**37. What is import and export**

**38. What is Destructing in the Javascript**

**39. Difference between Normal Functin and Arrow Function**

function addition(x, y) {

 return x + y;

}

const addition = (x, y) => x + y;

**41. Explain difference between map() and forEach()**

Use map() when you need to create a new array by applying a transformation to each element of the original array.

Use forEach() when you need to execute a function for each element of the array without modifying or creating a new array.

map() and forEach() are both methods used to iterate over arrays

|  |  |
| --- | --- |
| **forEach** | **Map** |
| Does not return a new array | Returns a new array with the transformed elements |
| Since forEach() returns undefined, it cannot be chained with other array methods. | Since map() returns a new array, it can be chained with other array methods like filter(), reduce(), etc. |

**42. What are events in Javascript**

events are actions or occurrences that happen in the browser.

* Mouse Events
* Keyboard Events
* Form Events etc..

**43. What is null and undefined in Javascript**

In JavaScript, both null and undefined represent the absence of a value

**undefined** is a primitive value that indicates that a variable has been declared but not yet assigned a value.

**null** is a primitive value that represents the intentional absence of any object value. a variable should have no value or that an object reference is empty.

**44. Explain the difference between == and ====**

type coercion => converting value from one type to other

1 == "1" => true => type coercion

1 === "1" => false => not do type coercion just check with data type