**Q-1 What is JavaScript? Explain the role of JavaScript in web development.**

**Ans :-**

Java Script is High level client side interpreted synchronous scripting programming language

* Use for : 1-website development

2-Mobile Application development

3- Game development

4-Scripting

🡪 role of JavaScript in web development:- JavaScript is a scripting language used to develop web pages. JS allows developers to create a dynamic and interactive web page to interact with visitors and execute complex actions. It also enables users to load content into a document without reloading the entire page

**Q-2 How is JavaScript different from other programming languages like Python or**

**Java?**

**Ans :-**

1. Interpreted vs. Compiled:

JavaScript: Interpreted, meaning code is executed line by line without a prior compilation step. This makes it more flexible and dynamic.

Python and Java: Compiled, requiring code to be translated into machine-readable format before execution. This generally results in faster performance but less flexibility.

2. Client-Side vs. Server-Side:

* JavaScript: Primarily a client-side language, running within web browsers to create interactive web pages. However, with Node.js, it can also be used for server-side development.

3. Prototype-Based Inheritance vs. Class-Based Inheritance:

* JavaScript: Uses prototype-based inheritance, where objects inherit properties and methods from other objects. This provides flexibility but can be less intuitive for those accustomed to class-based inheritance.
* Python and Java: Employ class-based inheritance, a more traditional approach where objects inherit from classes.

4. Functional Programming Support:

* JavaScript: Supports functional programming paradigms, allowing for writing concise and declarative code.
* Python and Java: While they support functional programming to varying degrees, JavaScript's emphasis on functional programming is more prominent.

**Q-3 What are variables in JavaScript? How do you declare a variable using var, let,**

**and const?**

**Ans:-**

Variables are like containers that store data values in a JavaScript program. They allow you to manipulate and reuse data throughout your code.

Let a=value

Var b=value

Const c=value

**Q-4 : Explain the different data types in JavaScript. Provide examples for each.**

**Ans :-**

🡪 Number: - Represents numeric values, both integers and floating-point numbers

Example: -

let age = 25; let pi = 3.14159;

🡪 String: - Represents a sequence of characters enclosed in single quotes, double quotes, or back-ticks.

Example: -

let name = "abc"; let greeting = 'Hello, world!';

🡪 Boolean: - Represents a logical value, either true or false.

Example: -

Let a=true; var b=false

🡪 Undefined:- Represents a variable that has been declared but not yet assigned a value

Example: -

let x;

console.log(x); Output: undefined

🡪 Null :- Represents an intentional absence of any object value.

Example :-

Let Num=null

**Q-5 What is the difference between undefined and null in JavaScript**

**Ans :-**  undefined means a variable has been declared but has not yet been assigned a value, whereas null is an assignment value, meaning that a variable has been declared and given the value of null .

Q-6 : What are the different types of operators in JavaScript? Explain with examples.

Ans :-

🡪Arithmetic operator :- Arithmetic operators are used to perform mathematical operations.

Examples;

Operator name example

+ Addition let sum=3+9

- Subtraction let result=5-2

\* Multiplication let result=6\*6

/ Division let result=200/10

% Modulus let result=123%10

++ Increment let result=10 result++

-- Decrement let result=10 result--

🡪Assignment operators are used to assign values to variables.

Examples: -

Operator name Example

= Assign let a=10

+= Add and Assign let a+=10

-= Minus and Assign let a-=10

\*= Multiple. Ans assign let a\*=10

/= Divide and Assign let a/=10

%= Modulus and Assign let a%=10

🡪 Comparison operators are used to compare values.

Operator Description Example

== Equal to 5 == 5

!= Not equal to 5 != 3

=== Strict equal to (value and type) 5 === "5"

!== Strict not equal to (value and type) 5 !== "5"

> Greater than 10 > 5

< Less than 5 < 10

>= Greater than or equal to 10 >= 10

<= Less than or equal to 5 <= 10

🡪 Logical operators are used to combine conditions.

Operator name example

&& And (a>10 && a< 30)

|| OR (a>10 || a< 30)

! NOT (!a>10)

**Q-7 What is the difference between == and === in JavaScript?**

**Ans :-**

== check only value and === check value + Data type

**Q-8 : What is control flow in JavaScript? Explain how if-else statements work with an**

**example.**

**Ans :-** Control flow in JavaScript is the order in which a computer runs code, from top to bottom. It is affected by statements that change the flow, such as loops, conditionals, and functions

Example:

Let a=10

If(a>=10){

Console.log(“If block Executed”)}

Else{

Console.log(“Else block Executetd”)

}

**Q-9 Describe how switch statements work in JavaScript.**

**Ans :-** A switch statement in JavaScript is a control flow statement that allows you to execute different code blocks based on the value of an expression. It's often used as an alternative to multiple if...else statements, especially when you need to compare a single value against multiple possible cases.

**Q-10 Explain the different types of loops in JavaScript (for, while, do-while). Provide a**

**basic example of each.**

**Ans :** There are 3 types of loop in js

1-for loop- It is entry control loop Used when you know the exact number of iterations in advance.

Example:-

For(let i=0;i<10;i++){

Console.log(i)

}

2- while loop :- while loop is entry control loop Used when you want to repeat a block of code as long as a condition is true.

Example:-

Let a=1

While(a<=10){

Console.log(a)

a++

}

3- Do while loop :- Do while loop is exist control loop Similar to a while loop but guarantees that the code inside the loop is executed at least once.

Example :

Let a=1

Do(

Console.log(a)

a++

)While(a<=10)

**Q=11 : What is the difference between a while loop and a do-while loop?**

**Ans :-**

* While loop

Evaluates the condition at the beginning of each iteration, before executing the loop body. This is called an entry controlled loop. If the condition is false, the loop body will not execute.

* Do-while loop

Executes the loop body once, then evaluates the condition at the end of each iteration. This is called an exit controlled loop. The loop body is guaranteed to execute at least once, regardless of the condition

**Q-12 What are functions in JavaScript? Explain the syntax for declaring and calling a**

**function.**

**Ans :-** A JavaScript function is a reusable block of code that performs a specific task. Functions are declared using the function keyword, followed by a name, parameters, and curly braces. To call a function, you write the function name followed by parentheses

Example:

Let Fun=()=>{

Console.log(“Function called…”)

}

Fun()

**Q-13 What is the difference between a function declaration and a function expression?**

**Ans :-** function declarations are named and can be called before they are defined, while function expressions are anonymous and must be defined before being called

**Q-14 : Discuss the concept of parameters and return values in functions.**

**Ans :-**

Parameter :

Parameters are variables that are passed to a function when it's called. They act as inputs to the function, allowing you to customize its behaviour based on different values.

When a function is defined, you specify the parameters within its parentheses.

Multiple parameters can be separated by commas.

Return value :-

A function can return a value using the return keyword.

The returned value can be used in other parts of your code.

A function can only return one value at a time

**Q-15 What is an array in JavaScript? How do you declare and initialize an array?**

**And :** array in JavaScript is a data structure that stores a collection of elements. These elements can be of various data types, such as numbers, strings, objects, or even other arrays.

Let art=[1,2,3,”Tops”,33,true,false]

**Q-16 Explain the methods push(), pop(), shift(), and unshift() used in arrays**

**Ans :-**

Push() :- push method use for add element in last index of the array

Pop() :- pop method use for remove last index of element in array

Shift() :- shift method use for remove first index element of the array

Upshift() :- upshift method use for add element first index in the array

**Q-17 What is an object in JavaScript? How are objects different from arrays?**

**Ans :** object in JavaScript is a collection of key-value pairs, where each key is a string and each value can be of any data type, including numbers, strings, Booleans, arrays, functions, or other objects. Objects are used to represent real-world entities or concepts with properties and behaviors.

**Q-18 Explain how to access and update object properties using dot notation and**

**bracket notation.**

**Ans :**

Let Obj ={

Name: “Tops”,

Age:22,

City: “Dhokla”

}

Using dot notation ----

Obj.name

Using bracket notation---

Obj[name]

Q-19 What are JavaScript events? Explain the role of event listeners.

Ans :- A JavaScript event is a specific action that occurs within a web page or application, such as clicking on an element, moving the mouse, pressing a key, or loading a page.

🡪 Event listeners detect and respond to events in a program or application. They allow developers to write code that reacts to specific user interactions or actions, such as: button clicks, mouse movements, keyboard input, and window sizing

**Q-20 : How does the addEventListener() method work in JavaScript? Provide**

**an example.**

**Ans :-** The method addEventListener() works by adding a function, or an object that implements a handle Event() function, to the list of event listeners for the specified event type on the Event Target on which it's called.

Example: -

    <div id="main"><h1>main</h1></div>

    <script>

        document.getElementById("main").addEventListener("click",()=>{

            alert("called")

        })

    </script>

**Q-21 : What is the DOM (Document Object Model) in JavaScript? How does JavaScript**

**interact with the DOM?**

**Ans :-** The DOM (Document Object Model) is a programming interface for web documents. It represents the page so that programs can manipulate its structure, style, and content. Essentially, the DOM is a tree-like structure where each node represents a part of the document, such as an element, attribute, or text.

document.getElementById('id')

document.getElementsByClassName('class')

document.getElementsByTagName('tag')

document.querySelector('selector')

document.querySelectorAll('selector')

**Q-22 : Explain the methods getElementById(), getElementsByClassName(), and**

**query Selector() used to select elements from the DOM**

**Ans :-**

1. getElementById()

This method is used to select an element by its ID.

let element = document.getElementById("id");

2. getElementsByClassName()

This method is used to select all elements with a specified class name.

let elements = document.getElementsByClassName("class Name");

3. query Selector()

The query Selector() method allows you to select the first element that matches a CSS selector.

let element = document.querySelector("selector");

**Q-23 Explain the set Timeout() and set Interval() functions in JavaScript. How are**

**they used for timing events?**

**Ans :-**

1. set Timeout()

The set Timeout() function is used to execute a function or code block once after a specified delay

setTimeout(function, delay);

2. set Interval()

The set Interval() function is used to execute a function or code block repeatedly at fixed time intervals

set Interval(function, interval);

Q-24 Provide an example of how to use set Timeout () to delay an action by 2 seconds.

Ans :-

 <script>

        setTimeout (() => {

            document. Write('<br>Called')

        }, 2000);

    </script>

**Q-25 What is error handling in JavaScript? Explain the try, catch, and finally**

**blocks with an example.**

**Ans :-** Error handling in JavaScript is the process of catching and managing errors that occur during the execution of code. This is crucial for maintaining the reliability and stability of your application.

function divide(a, b) {

try {

if (b === 0) {

throw new Error("Cannot divide by zero");

}

console.log(a / b)

} catch (error) {

console.log("Error: " + error. message)

} finally {

console.log("Execution completed.")

}

}

divide(10, 0)

divide(10, 2)

Q-26 : Why is error handling important in JavaScript applications?

Ans :- Error handling is crucial in programming languages to ensure the smooth execution of code and maintain system stability. In more detail, error handling is a fundamental aspect of programming that allows developers to anticipate potential problems and address them proactively.

- important

Prevents crashes

Improves user experience

Simplifies debugging

Enhances security

Reduces technical debt