**Module (JAVASCRIPT BASIC & DOM) – 4**

**(Basic logic Question)**

Q.1 What is JavaScript. How to use it?

Ans JavaScript is a versatile programming language primarily used for creating dynamic and interactive elements on web pages. Originally designed to run in web browsers, it has now extended its reach to server-side development, desktop applications, and even mobile app development through frameworks like React Native.

Q.2 How many type of Variable in JavaScript?

Ans A JavaScript variable is simply a name of storage location. There are two types of variables in JavaScript : local variable and global variable. There are some rules while declaring a JavaScript variable (also known as identifiers). Name must start with a letter (a to z or A to Z), underscore( \_ ), or dollar( $ ) sign.

Q.3 Define a Data Types in js?

Ans (1)Primitive Data Types:

String: Represents text data enclosed within single ('') or double ("") quotes.

Number: Represents numeric data, including integers and floating-point numbers.

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

Null: Represents the intentional absence of any value.

Undefined: Represents a variable that has been declared but has not been assigned a value.

Symbol: Introduced in ECMAScript 6 (ES6), represents a unique identifier.

(2)Composite Data Types:

Object: Represents a collection of key-value pairs, where keys are strings (or symbols) and values can be any data type, including other objects.

Array: Represents an ordered collection of values, where each value can be of any data type, and elements are accessed by their numerical index.

Function: Represents a reusable block of code that can be invoked with zero or more arguments.

Q.4 Write a mul Function Which will Work Properly When invoked With Following Syntax.

Ans The MUL function is a miniature of the multiplication function. In this function, we call the function that required an argument as a first number, and that function calls another function that required another argument and this step goes on.

Q.5 What the deference between undefined and undeclare in JavaScript?

Ans 1. Undefined: In JavaScript, "undefined" is a special value that represents the absence of a value. It is a primitive data type that indicates that a variable has been declared but has not been assigned a value, or a function has been declared but has not been defined. For example:

javascript

Copy code

let x;

console.log(x

function foo() {

}

console.log(foo());

In the above example, the variable x is declared but not assigned a value, so its value is undefined. Similarly, the function foo is declared but has no return statement, so when it is called, it returns undefined.

2. Undeclared: "Undeclared" refers to a situation where a variable is used without being declared using the var, let, or const keyword. In such cases, JavaScript will throw a ReferenceError because it does not recognize the variable. For example:

javascript

Copy code

console.log(y);

In the above example, y is used without being declared, so JavaScript throws a ReferenceError because it cannot find any declaration for the variable y.

Q.6 Using console.log() print out the following statement: The quote 'There is no exercise better for the heart than reaching down and lifting people up.' by John Holmes teaches us to help one another. Using console.log() print out the following quote by Mother Teresa:

Ans console.log("The quote 'There is no exercise better for the heart than reaching down and lifting people up.' by John Holmes teaches us to help one another.");

console.log("The quote 'If you can't feed a hundred people, then feed just one.' by Mother Teresa teaches us to help those in need.");

Q.7 Check if typeof '10' is exactly equal to 10. If not make it exactly equal?

Ans if (typeof '10' !== typeof 10) {

let num = Number('10');

console.log(num); // Output: 10

} else {

console.log("The types are exactly equal.");

}

Q.8 Write a JavaScript Program to find the area of a triangle?

Ans function calculateTriangleArea(base, height) {

return (base > 0 && height > 0) ? (base \* height) / 2 : "Base and height must be positive numbers.";

}

let base = 5; // Base of the triangle

let height = 8; // Height of the triangle

let area = calculateTriangleArea(base, height);

console.log("The area of the triangle with base", base, "and height", height, "is:", area);

Q.9 Write a JavaScript program to calculate days left until next Christmas?

ans

function daysUntilNextChristmas() {

let currentDate = new Date();

let currentYear = currentDate.getFullYear();

let christmasDate = new Date(currentYear, 11, 25); // Month is zero-based (0 = January, 11 = December)

if (currentDate > christmasDate) {

christmasDate.setFullYear(currentYear + 1);

}

let timeDiff = christmasDate.getTime() - currentDate.getTime();

let daysLeft = Math.ceil(timeDiff / (1000 \* 3600 \* 24));

return daysLeft;

}

let daysLeftUntilChristmas = daysUntilNextChristmas();

console.log("There are", daysLeftUntilChristmas, "days left until next Christmas.");

Q.10 What is Condition Statement?

Ans Conditional statements in programming are used to control the flow of a program based on certain conditions. These statements allow the execution of different code blocks depending on whether a specified condition evaluates to true or false, providing a fundamental mechanism for decision-making in algorithms. In this article, we will learn about the basics of Conditional Statements along with their different types.

Q.11 Find circumference of Rectangle formula : C = 4 \* a ?

Ans function rectangleCircumference(side) {

return 4 \* side;

}

let sideLength = 5; // Length of one side of the rectangle

let rectangleCircumferenceValue = rectangleCircumference(sideLength);

console.log("The 'circumference' of the rectangle is:", rectangleCircumferenceValue);

Q.12 WAP to convert years into days and days into years?

Ans // Function to convert years to days

function yearsToDays(years) {

// Assuming a non-leap year for simplicity (365 days)

return years \* 365;

}

// Function to convert days to years

function daysToYears(days) {

// Assuming a non-leap year for simplicity (365 days)

return days / 365;

}

// Example usage:

let years = 5;

let days = 1825; // Corresponds to approximately 5 years

// Convert years to days

let daysConverted = yearsToDays(years);

console.log(years + " years is equal to " + daysConverted + " days.");

// Convert days to years

let yearsConverted = daysToYears(days);

console.log(days + " days is equal to approximately " + yearsConverted.toFixed(2) + " years.");

Q.13 Convert temperature Fahrenheit to Celsius? (Conditional logic Question)

Ans // Function to convert temperature from Fahrenheit to Celsius

function fahrenheitToCelsius(fahrenheit) {

let celsius = (5 / 9) \* (fahrenheit - 32);

return celsius;

}

let fahrenheitTemp = 32;

let celsiusTemp = fahrenheitToCelsius(fahrenheitTemp);

console.log(fahrenheitTemp + " degrees Fahrenheit is equal to " + celsiusTemp.toFixed(2) + "degrees Celsius.");

Q.15 What is the result of the expression (5 > 3 && 2 < 4)?

Ans the result of the expression (5 > 3 && 2 < 4) is true.

Q.16 What is the result of the expression (true && 1 && "hello")?

Ans the result of the expression (true && 1 && "hello") is "hello".

Q.17 What is the result of the expression true && false || false && true?

Ans the result of the expression true && false || false && true is false.

Q.18 What is a Loop and Switch Case in JavaScript define that ?

Ans a loop is a programming construct that allows you to execute a block of code repeatedly until a specified condition is met.

in JavaScript used to perform different actions based on different conditions. It's an alternative to using multiple if-else statements, especially when there are multiple possible conditions to check

Switch case evaluates an expression and compares it against multiple cases. If a case matches, the corresponding block of code is executed. The break statement is used to exit the switch block after a case is matched. If no case matches, the code inside the default block is executed (if it's provided).

Q.19 What is the use of is Nan function?

Ans The isNaN() function in JavaScript is used to determine whether a value is NaN (Not-a-Number) or not.

NaN is a special value in JavaScript that represents a value which is "not a number". It typically arises as a result of a mathematical operation that cannot produce a meaningful numeric result. For example, dividing zero by zero or attempting to convert a non-numeric string to a number can result in NaN.

Q.20 What is the difference between && and || in JavaScript?

ans Logical AND (&&):

The logical AND operator (&&) returns true if both operands are true, otherwise it returns false.

If the first operand evaluates to false, the second operand is not evaluated because the result will always be false regardless of its value.

Example: true && true evaluates to true, true && false evaluates to false.

Logical OR (||):

The logical OR operator (||) returns true if at least one of the operands is true, otherwise it returns false.

If the first operand evaluates to true, the second operand is not evaluated because the result will always be true regardless of its value.

Example: true || false evaluates to true, false || false evaluates to false.

Q.21 What is the use of Void (0)?

The use of void(0) in JavaScript is primarily as a way to create an expression that returns undefined. It's commonly used in situations where you want to execute a JavaScript statement without navigating away from the current page. For example, in an HTML anchor tag (<a>), you might see href="javascript:void(0)" to prevent the browser from following the link when it's clicked.

Q.30 What are the looping structures in JavaScript? Any one Example?

Ans javaScript provides several looping structures for iterating over data or executing code repeatedly. Some of the common looping structures in JavaScript include:

1. for loop: Executes a block of code a specified number of times.

for (let i = 0; i < 5; i++) {

console.log("Iteration", i);

}

1. while loop: Executes a block of code while a specified condition is true.

let i = 0;

while (i < 5) {

console.log("Iteration", i);

i++;

}

1. do-while loop: Similar to a while loop, but the code block is executed at least once, even if the condition is false.

let i = 0;

do {

console.log("Iteration", i);

i++;

} while (i < 5);

Q.47,50 What is the drawback of declaring methods directly in JavaScript objects?

Ans Declaring methods directly in JavaScript objects can lead to several drawbacks, especially as the complexity of your codebase grows:

1.Namespace Pollution: If you're declaring methods directly within an object, especially if it's a global or widely shared object, you risk polluting the global namespace. This can lead to naming conflicts with other parts of your code or third-party libraries.

2.Code Maintainability: Declaring methods directly within objects can make your code harder to maintain, especially if you have a large number of methods or complex logic within them. It can become difficult to track down where a particular method is defined, leading to code that's harder to debug and refactor.

3.Inflexibility: Objects with directly declared methods can be less flexible and reusable. You might find it harder to extend or modify the behavior of these methods without directly modifying the object itself, which can introduce unintended side effects or break other parts of your code.

4.Readability and Separation of Concerns: Mixing method definitions with object properties can make your code less readable and violate the principle of separation of concerns. It's often clearer to define methods separately from object instantiation, especially if those methods perform complex operations or interact with external resources.

Q.57 how many type of JS Event? How to use it?

Ans 1.Mouse Events: These events are triggered by user interactions with the mouse, such as clicks, mouseovers, and mouseouts.

2.Keyboard Events: These events are triggered by user interactions with the keyboard, such as key presses and releases.Example:

3.Form Events: These events are triggered by user interactions with HTML forms, such as submitting a form or changing the value of an input field.

4. Window Events: These events are triggered by actions related to the browser window, such as loading, resizing, or closing the window.

5. Document Events: These events are triggered by actions related to the HTML document, such as loading or unloading the document.

6. event Delegation: This is a technique where you attach a single event listener to a parent element, and then use event delegation to handle events that occur on its descendant elements. This is useful for handling events on dynamically created elements or a large number of elements.

Q.59 What is Bom vs Dom in JS?

Ans 1.DOM (Document Object Model):

The Document Object Model represents the structure of an HTML or XML document as a tree-like structure where each node represents a part of the document, such as elements, attributes, and text.

The DOM provides a standardized way for JavaScript to interact with and manipulate the structure, content, and style of web documents.

JavaScript can access and modify elements and attributes within the DOM using methods and properties provided by the DOM API.

Examples of DOM manipulation include adding or removing elements, changing element content or attributes, and responding to user events.

2.BOM (Browser Object Model):

The Browser Object Model represents various aspects of the web browser itself, such as the browser window, history, location, and screen.

Unlike the DOM, the BOM is not standardized and can vary between different browsers.

The BOM provides interfaces for controlling browser behavior, such as navigating to a new URL, opening new browser windows or tabs, and interacting with browser features like the history and cookies.

• What is negative Infinity?

Ans Negative Infinity, denoted as -Infinity, is a special value in JavaScript representing the negative infinity. It is the opposite of positive infinity (Infinity). Both positive and negative infinity are values of the Number type in JavaScript.

• Which company developed JavaScript?

Ans JavaScript was developed by Netscape Communications Corporation

• What are undeclared and undefined variables?

Ans Undeclared variables: These are variables that have not been declared using a var, let, or const keyword before they are used. Accessing an undeclared variable typically results in a ReferenceError in JavaScript.

Undefined variables: These are variables that have been declared but have not been assigned a value, or variables that have been assigned the special value undefined. Accessing an undefined variable does not result in an error, but the value returned will be undefined.

• What is the difference between ViewState and SessionState?

Ans

| **Feature** | **ViewState** | **SessionState** |
| --- | --- | --- |
| Scope | Limited to a single page | Available across multiple pages within a session |
| Storage | Stored in a hidden field on the page | Stored on the server-side |
| Persistence | Persists only for the current page | Persists for the entire session |
| Size | Typically smaller in size | Can store larger amounts of data |
| Performance | Faster because it's client-side | Slower because it involves server communication |
| Security | Data is visible to the client | Data is stored securely on the server |
| Usage Scenario | Used for retaining page-specific data | Used for storing user-specific data throughout the session |

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

Ans 1.Changing Inline Styles:

You can directly modify the inline styles of an element using the style property.

2. Adding or Removing CSS Classes:

You can add or remove CSS classes to an element using the classList property.

3. Setting Class Name Directly:

You can also directly set the className property of an element to change its classes.

• How to read and write a file using JavaScript?

Ans In JavaScript, you can't directly access files on the user's machine due to browser security restrictions. However, you can interact with files using the File API in modern browsers or by using Node.js on the server-side.

• What is the function of the delete operator?

Ans In JavaScript, the **delete** operator is used to remove a property from an object. It is also used to remove an element from an array, although it leaves a gap in the array.

It's important to note that when you use delete on an array element, the element itself is removed, but the length of the array remains the same, and a gap is left at the deleted index. So, you may get unexpected results when iterating over arrays using methods like forEach() or checking its length.

• What are all the types of Pop up boxes available in JavaScript?

Ans In JavaScript, there are several types of pop-up boxes that you can use to interact with users. These pop-up boxes are created using built-in functions:

Alert Box: Displays a message to the user with an OK button.

Confirm Box: Asks the user to confirm or cancel an action.

Prompt Box: Prompts the user to input text.

• How can a page be forced to load another page in JavaScript?

Ans In JavaScript, you can force a page to load another page by setting the window.location property to the URL of the page you want to load.

You can also use other properties of window.location to control navigation, such as window.location.assign() or window.location.replace(), but window.location.href is commonly used for this purpose.

• What are the disadvantages of using innerHTML in JavaScript?

Ans While innerHTML is a convenient and widely used property in JavaScript for manipulating HTML content within elements, it does have some disadvantages:

Security Risks: Using innerHTML to directly insert HTML content into a page can expose your application to Cross-Site Scripting (XSS) attacks if the content is not properly sanitized. If the inserted content contains executable scripts, they will be executed in the context of your web page.

Performance Overhead: When you set innerHTML, the browser has to re-parse and re-render the entire HTML content of the element, which can be inefficient for large documents or frequent updates. This can lead to performance degradation, especially on mobile devices or older browsers.

Event Handlers: If the content you're inserting using innerHTML contains elements with event handlers attached to them, those event handlers will be detached and lost. This can cause unexpected behavior if you rely on event handling for those elements.

Accessibility: Manipulating the DOM using innerHTML can lead to accessibility issues if the changes you make are not properly handled or if they interfere with screen readers or other assistive technologies.

Limited Control: When you use innerHTML, you're essentially replacing the entire contents of an element. This can be problematic if you need to make more granular changes or if you want to preserve existing content or state.

Vendor-Specific Behaviors: Different browsers may have different behaviors or quirks when using innerHTML, which can lead to inconsistent results or unexpected behavior across browsers.