JAVASCRIPT ASSIGNMENT

Q 1 = What is JavaScript?

Ans = *JavaScript* was initially created to "make web pages alive".

The programs in this language are called *scripts*. They can be written right in a web page's HTML and run automatically as the page loads.

Scripts are provided and executed as plain text. They don't need special preparation or compilation to run.

In this aspect, JavaScript is very different from another language called Java.

Today, JavaScript can execute not only in the browser, but also on the server, or actually on any device that has a special program called the JavaScript engine.

The browser has an embedded engine sometimes called a "JavaScript virtual machine".

Different engines have different "codenames". For example:

- V8 – in Chrome, Opera and Edge.
- SpiderMonkey – in Firefox.

Q 2 = What is the use of is NaN function?

Ans = The JavaScript **is NaN()** Function is used to check whether a given value is an illegal number or not. It returns true if the value is a NaN else returns false. It is different from the Number is NaN() Method.

Q 3= What is negative Infinity?

Ans = The **negative infinity** in JavaScript is a constant value that is used to represent a value that is the lowest available. This means that no other number is lesser than this value. It can be generated using a self-made function or by an arithmetic operation.

Q 4 = Which company developed JavaScript?

Ans = The JavaScript Programming Language was developed by Brendan Eich when he was an employee of Netscape. Brendan Eich is also, the founder of Mozilla Foundation.

Q 5 = What are undeclared and undefined variables?

Ans = **Undefined:** It occurs when a variable has been declared but has not been assigned any value. Undefined is not a keyword.

Undeclared:
It occurs when we try to access any variable that is not initialized or declared earlier using the *var* or *const keyword*. If we use *'typeof'* operator to get the value of an undeclared variable, we will face the *runtime error* with the return value as **"undefined"**. The scope of the undeclared variables is always global.

Q 6 = Write the code for adding new elements dynamically?

Ans = Javascript is a very important language when it comes to learning how the browser works. Often there are times we would like to add dynamic elements/content to our web pages. This post deals with all of that.

Creation of new element: New elements can be created in JS by using the **createElement()** method.
Syntax:
document.createElement("<*tagName*>");
// Where <*tagName*> can be any HTML
// tagName like div, ul, button, etc.

// newDiv element has been created
For Eg: **let newDiv = document.createElement("div");**

Q 7 = What is the difference between ViewState and SessionState?

Ans = **ViewState:** It is maintained at only one level that is page-level. Changes made on a single page is not visible on other pages. Information that is gathered in view state is stored for the clients only and cannot be transferred to any other place. View state is synonymous with serializable data only.
ViewState has a tendency for the persistence of page-instance-specific data. When view state is used, the values posted of a particular page persist in the browse area that the client is using and post back only when the entire operation is done. The data of the previous page is no longer available when another page is loaded. Also, Data is not secure in this case because it is exposed to clients. Encryption can be used for data security.

SessionState:
It is maintained at session-level and data can be accessed across all pages in the web application. The information is stored within the server and can be accessed by any person that has access to the server where the information is stored.
SessionState has the tendency for the persistence of user-specific data and is maintained on the server-side. This data remains available until the time that the session is completed or the browser is closed by the user. The session state is only valid for type objects.

Q 8 = What is === operator?

Ans : === (Triple equals) is a strict equality comparison operator in JavaScript, which returns false for the values which are not of a similar type. This operator performs type casting for equality. If we compare 2 with "2" using ===, then it will return a false value.

Q 9 = How can the style/class of an element be changed?

Ans = We can change, add or remove any CSS property from an HTML element on the occurrence of any event with the help of JavaScript. There are two common approaches that allow us to achieve this task.

- style.property
- Changing the class itself

Approach 1: Changing CSS with the help of the style property:

Syntax:
document.getElementById("id").style.property = new_style

Example: In this example, we have built a PAN number validator. First, we will take the input value and match it with a regex pattern. If it matches then using JavaScript add an inline style on the <p> tag. Otherwise, add a different style on the <p> tag.

Approach 2: Changing the class itself – We can use two properties that can be used to manipulate the classes.
The classList Property: The **classList** is a read-only property that returns the CSS class names of an element as a DOMTokenList object.
Syntax:
document.getElementById("id").classList

You can use the below-mentioned methods to add classes, remove classes, and toggle between different classes respectively.

- **The add() method:** It adds one or more classes.
- **The remove() method:** It removes one or more classes.
- **The toggle() method:** If the class does not exist it adds it and returns true. It removes the class and returns false. The second boolean argument forces the class to be added or removed.

Q10= How to read and write a file using JavaScript?

Ans= The read and write operations in a file can be done by using some commands. But the module which is required to perform these operations is to be imported. The required module is 'fs' which is called as File System module in JavaScript.

Write operation on a file

After the File System file is imported then, the writeFile() operation is called. The writeFile() method is used to write into the file in JavaScript. The syntax of this method is as follows –

```
writeFile(path,inputData,callBackFunction)
```

The writeFile() function accepts three parameters –

- **Path** – The first parameter is the path of the file or the name of the file into which the input data is to be written.
  If there is a file already, then the contents in the file are deleted and the input which is given by the user will get updated or if the file is not present, then the file with that will be created in the given path and the input information is written into it.
- **inputData** – The second parameter is the input data which contains the data to be written in the file that is opened.
- **callBackFuntion** – The third parameter is the function which is the call back function which takes the error as the parameter and shows the fault if the write operation fails.

Reading from the file

After the File System module is imported, the reading of the file in JavaScript can be done by using the readFile() function.

Syntax

The syntax to read from a file is as follows –

```
readFile(path, format, callBackFunc)
```

The readFile() function accepts three parameters including one optional parameter.

- **Path** – The first parameter is the path of the test file from which the contents are to read. If the current location or directory is the same directory where the file which is to be opened and read is located then, only the file name has to be given.
- **Format** – The second parameter is the optional parameter which is the format of the text file. The format can be ASCII, utf-8 etc.
- **CallBackFunc** – The third parameter is the call back function which takes the error as the parameter and displays the fault is any raised due to the error.

Q11= What are all the looping structures in JavaScript?

Ans = Loops are used in JavaScript to perform repeated tasks based on a condition. Conditions typically return true or false. A loop will continue running until the defined condition returns false.

There are mainly two types of loops:

- **Entry Controlled loops**: In this type of loop, the test condition is tested before entering the loop body. **For Loop** and **While Loops** are entry-controlled loops.

- **Exit Controlled Loops**: In this type of loop the test condition is tested or evaluated at the end of the loop body. Therefore, the loop body will execute at least once, irrespective of whether the test condition is true or false. the **do-while loop** is exit controlled loop.

For loop:

The for Loop The 'for' loop is the most compact form of looping. It includes the following three important parts:

- The loop initialization where we initialize our counter to a starting value. The initialization statement is executed before the loop begins.

- The test statement which will test if a given condition is true or not. If the condition is true, then the code given inside the loop will be executed, otherwise the control will come out of the loop.

- The iteration statement where you can increase or decrease your counter. You can put all the three parts in a single line separated by semicolons.

Syntax: The syntax of for loop is JavaScript is as follows:

```
for (initialization; test condition; iteration statement){

        Statement(s) to be executed if test condition is true

}
```

While Loop

A **While Loop** in Javascript is a control flow statement that allows the code to be executed repeatedly based on the given boolean condition. The while loop can be thought of as a repeating if statement.

Syntax:

```
while (condition) {
    // Statements
}
```

do… while loop

A **do… while loop** in JavaScript is a control statement in which the code is allowed to execute continuously based on a given boolean condition. It is like a repeating if statement.
The do…while loop can be used to execute a specific block of code at least once

Syntax:

```
do {
    // Statements
}
while(conditions)
```

Q12 = What is the function of the delete operator?

Ans= Delete is comparatively a lesser-known operator in JavaScript. This operator is more specifically used to delete JavaScript object properties.

The JavaScript **pop()**, **shift()**, or **splice()** methods are available to delete an element from an array. But because of the key-value pair in an object, deleting is more complicated. Note that, the delete operator only works on objects and not on variables or functions.

Syntax:

```
delete object

// or

delete object.property

// or

delete object['property']
```

Q 13 = What are all the types of Pop up boxes available in JavaScript?

Javascript Popup Box

There is three type of popup box in javascript.

1. Alert box
2. Confirm Box
3. Prompt Box

1) Alert Box

An alert box is often used if you want to make sure information comes through to the user. When an alert box pops up, the user will have to click "OK" to proceed.

2) Confirm Box

A confirm box is often used if you want the user to verify or accept something. When a confirm box pops up, the user will have to click either "OK" or "Cancel" to proceed. If the user clicks "OK", the box returns true. If the user clicks "Cancel", the box returns false.

3) Prompt Box

A prompt box is often used if you want the user to input a value before entering a page. When a prompt box pops up, the user will have to click either "OK" or "Cancel" to proceed after entering an input value. If the user clicks "OK" the box returns the input value. If the user clicks "Cancel" the box returns null.

Q=14 How can you convert the string of any base to an integer in JavaScript?

Ans= A method (or a function) provided by JavaScript called as **parseInt()**.
This is a special method, provided by JavaScript, that takes an integer value (of any base which is either specified or not) and further converts the string into an integer value.

Syntax:

parseInt(string_value, base)

Q=15 What is the function of the delete operator?

Ans = Delete is comparatively a lesser-known operator in JavaScript. This operator is more specifically used to delete JavaScript object properties.

Syntax:

delete object.property

delete object[property]

Q 16 = What is the use of Void (0)?

Ans= This Void operator allows evaluating expressions that produce a value into places where an expression that evaluates to undefined is desired.

The void operator is often used merely to obtain the undefined primitive value, usually using void(0) (which is equivalent to void 0). In these cases, the global variable undefined can be used.

It should be noted that the precedence of the void operator should be taken into account and that parentheses can help clarify the resolution of the expression following the void operator.

Syntax:

void expression

Q 17= How can a page be forced to load another page in JavaScript?

Ans = We can use ***window.location*** property inside the *script* tag to forcefully load another page in Javascript. It is a reference to a Location object that is it represents the current location of the document. We can change the URL of a window by accessing it.
Syntax:

<script>

window.location = <Path / URL>

</script

Q 18 = What are the disadvantages of using innerHTML in JavaScript?

Ans= The innerHTML property is a part of the Document Object Model (DOM) that is used to set or return the HTML content of an element. Where the return value represents the text content of the HTML element. It allows JavaScript code to manipulate a website being displayed. More specifically, it sets or returns the HTML content (the inner HTML) of an element. The innerHTML property is widely used to modify the contents of a webpage as it is the easiest way of modifying DOM. But there are some disadvantages to using innerHTML in JavaScript.

Disadvantages of using innerHTML property in JavaScript:

- **The use of innerHTML very slow:** The process of using innerHTML is much slower as its contents as slowly built, also already parsed contents and elements are also re-parsed which takes time.
-
- **Preserves event handlers attached to any DOM elements:** The event handlers do not get attached to the new elements created by setting innerHTML automatically. To do so one has to keep track of the event handlers and attach it to new elements manually. This may cause a memory leak on some browsers.
- **Content is replaced everywhere:** Either you add, append, delete or modify contents on a webpage using innerHTML, all contents is replaced, also all the DOM nodes inside that element are reparsed and recreated.