MODULE: 4 (JavaScript Basic & DOM)

1. What is JavaScript?

Ans. It is an **interpreted programming language** with object-oriented capabilities. JavaScript was first known as Livescript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java.

2. What is the use of is NaN function?

Ans. In JavaScript, **NaN** stands for **N**ot **a N**umber. It represents a value which is not a valid number. It can be used to check whether a number entered is a valid number or not a number. To assign a variable to NaN value, we can use one of the two following ways.

```
var monthNumber = 14;

if (monthNumber < 1 || monthNumber > 12) {

    // Assigning monthNumber NaN as
    // month number is not valid
    monthNumber = Number.NaN;

    console.log("Month number should be"
                + " between 1 and 12");
}
else {
    console.log(monthNumber);
}
```

3. What is negative Infinity?

Ans. Negative infinity is a **number in java script**, which is derived by 'dividing negative number by zero'. - A number object needs not to be created to access this static property.

4. Which company developed JavaScript?

Ans. Netscape :- Is the name of the company which developed the javascript the founder of the javascript language is Brenden eich The photograph of the founder is shown below. According to wikipedia Brendan Eich (/ˈaɪk/; born July 4, 1961) is an American technologist and creator of the JavaScript programming language. He co-founded the Mozilla project, the Mozilla Foundation and the Mozilla Corporation, and served as the Mozilla Corporation's chief technical officer and briefly, as its chief executive officer.

5. What are undeclared and undefined variables?

Ans. **Undefined:** It occurs when a variable has been declared but has not been assigned with 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 *var* or *const keyword*. If we use *'typeof'* operator to get the value of an undeclared variable, we will face the *runtime error* with return value as **"undefined"**. The scope of the undeclared variables is always global. **For example:**

Undefined:

```
var geek;
undefined
console.log(geek)
```

Undeclared:

```
//ReferenceError: myVariable is not defined
console.log(myVariable)
```

6. Write the code for adding new elements dynamically?

Ans.
```
<html>
<head> <title>t1</title>
<script type="text/javascript">
function addNode() { var newP = document.createElement("p");
var textNode = document.createTextNode(" This is a new text node");
newP.appendChild(textNode);
document.getElementById("firstP").appendChild(newP); }
</script> </head>
```

```
<body> <p id="firstP">firstP<p> </body>
</html>
```

7. What is the difference between ViewState and SessionState?

Ans. The basic difference between these two is that the **ViewState is to manage state at the client's end**, making state management easy for end-user while SessionState manages state at the server's end, making it easy to manage content from this end too. ViewState: It is maintained at only one level that is page-level.

8. What is === operator?

Ans. === is **Strict equal to**: `true` if the operands are equal and of the same type

```
For example: strict equal operator
console.log(2 === 2); // true
console.log(2 === '2'); // false
```

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

Ans. Select the element whose style properties needs to be change.

- Use element.style property to set the style attribute of an element.
- Set the properties either by using bracket notation or dash notation.

10. How to read and write a file using JavaScript?

Ans. The fs.readFile() and rs.writeFile() methods are used to read and write of a file using javascript. The file is read using the fs.readFile() function, which is an inbuilt method. This technique reads the full file into memory and stores it in a buffer.

11. What are all the looping structures in JavaScript?

Ans. **JavaScript supports different kinds of loops:**

- for - loops through a block of code a number of times
- for/in - loops through the properties of an object
- for/of - loops through the values of an iterable object
- while - loops through a block of code while a specified condition is true
- do/while - also loops through a block of code while a specified condition is true

12. How can you convert the string of any base to an integer in JavaScript?

Ans. the syntax that a user may use to convert a string into an integer value (of any base)- parseInt (string_value, base)
Alternatively, if we don't want to specify the base value and just want to convert our string value into an integer value itself, then we may use the following syntax also- parseInt (string_value)

13. What is the function of the delete operator?

Ans. The `delete` **operator** in JavaScript is used to delete an object's property.

If it is used to delete an object property that already exists, it returns `true` and removes the property from the object. However, deleting an object property that doesn't exist will not affect the object, but will still return `true`.

The only time `false` will be returned is when the `delete` operator is used to delete a variable or a function.

14. What are all the types of Popup boxes available in JavaScript?

Ans. There are total three types of boxes in javascript: alert box, confirm box and prompt box.

15. What is the use of Void (0)?

Ans. The javascript:void (0) can be used when we don't want to refresh or load a new page in the browser on clicking a hyperlink. We can use the operand 0 in two ways that are void (0) or void 0. Both of the ways work the same. The JavaScript :void (0) tells the browser to "do nothing" i.e., prevents the browser from reloading or refreshing the page.

16. 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.

```
<!DOCTYPE html>
<html lang="en">

<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible"
content="IE=edge">
<meta name="viewport" content=
"width=device-width, initial-scale=1.0">
</head>

<body>
<h3>This is the original page</h3>
<br>

<button onclick="force_load_gfg()">
```

17. What are the disadvantages of using inner HTML in JavaScript?

Ans. 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.

Appending to innerHTML is not supported: Usually, += is used for appending in JavaScript. But on appending to an Html tag using innerHTML, the whole tag is re-parsed.

Example:

```
<p id="geek">Geeks</p>
title = document.getElementById('#geek')

// The whole "geek" tag is reparsed
title.innerHTML += '<p> forGeeks </p>'
```

Old content replaced issue: The old content is replaced even if object.innerHTML = object.innerHTML + 'html' is used instead of object.innerHTML += 'html'. There is no way of appending without reparsing the whole innerHTML. Therefore, working with innerHTML becomes very slow. String concatenation just does not scale when dynamic DOM elements need to be created as the plus' and quote openings and closings becomes difficult to track.

Can break the document: There is no proper validation provided by innerHTML, so any valid HTML code can be used. This may break the document of JavaScript. Even broken HTML can be used, which may lead to unexpected problems.

Can also be used for Cross-site Scripting(XSS): The fact that innerHTML can add text and elements to the webpage, can easily be used by malicious users to manipulate and display undesirable or harmful elements within other HTML element tags. Cross-site Scripting may also lead to loss, leak and change of sensitive information.