JAVASCRIPT QUESTIONS - 3

1. What is a variable in JavaScript?

- **Variables are containers for storing data values in Java, There are different types of variables. For example: String can store text, such as "Hi". Integer can store whole numbers without decimals, such as 123.One java variable can hold only single type of data. Before using g a variable while programming, it is necessary to declare a variable. It mean to assign data to particular memory and use a name for that memory.**

2. How do you declare a variable in JavaScript?

- **In JavaScript, variables can be declared using a keywords like var, let or const, each keyword used in some specific conditions. Understanding these declarations help for unintended side effects in code.**
  - **JavaScript var**

    This keyword is used to declare variables globally and changeable also. It is good for a short length of codes, if the codes get huge then you will get confused.

    Eg: var X= 10;
  - **JavaScript let**

    This Keyword is used to declare variables locally. If you use this keyword to declare a variable can accessible locally and it is changeable as well. It is good if the code gets huge.

    Eg: Let name ="Shalomitha";
  - **JavaScript const**

    It can declare locally. If you use this keyword to declare a variable then the variable will only be accessible within that block similar to the variable define by using let and difference between let and const is that the variable define by using let and difference let and const is that the variable declare using const values can't be reassigned so we should assign the value while declaring the variable.

    Eg: Const Z=10;

3. What are the difference between var, let and const?

JavaScript var	JavaScript let	JavaScript const
Can be re declared	Cannot be re declared	Cannot be re declared
Can be reassigned a value	Can be reassigned a value	Cannot reassign the value
Only have global and function scope	Can have a block scope	Can have a block scope
Variables are hoisted on top and can be used anywhere	Variables must be initialized before use	Variables must be initialized before use
Can be re declared anywhere in the program	Can be re declared inside a block	Can never be re declared

4. Explain the variable hoisting in JavaScript?

- **Variable hoisting in JavaScript refers to the behavior where variable and function declarations are moved to the top of their containing scope during the compilation phase. This allows variables and functions to be used before their actual declarations in the code.**
- **hoisting moves declarations to the top of their scope, allowing early usage within the scope, but variables declared with `let` and `const` are not initialized until the declaration is encountered.**

5. What are the scoping rules for var, let and const?

var:

- **Function-scoped or global-scoped.**
- **Not block-scoped.**
- **Hoisted and initialized with undefined.**

let:

- **Block-scoped.**
- **Hoisted but not initialized (temporal dead zone).**

Const:

- **Block-scoped.**
- **Hoisted but not initialized (temporal dead zone).**
- **Cannot be reassigned once initialized.**

6. How can you use the template literals in JavaScript?

- **We can use template literals to embed JavaScript expressions or variables with the help of the ${...} syntax.**
  For example,

```
let number1= 8;

let number2= 3;

let result= 'The sum of ${number} and ${number2} is ${number1+Number2}

console.log (result);

Output: The sum of 8 and 3 is 11.
```

7. List the primitive data types in JavaScript?

- **String**
- **Number**
- **Boolean**
- **BigInt**
- **Null**
- **Undefined**
- **Symbol**

8. What are the difference between Null and Undefined?

- **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.**

- **Both null and undefined represent an absence of value, null is an assignment value indicating intentional absence, whereas undefined is a default value indicating that a variable has not been initialized or a property does not exist.**

9. How do you check a type of variable in JavaScript?

- **'typeof' is a keyword in JavaScript will return the type of variable when you call it and also this can use to validate function parameters or check if variable are defined. The type operator is useful because it is an easy way to check the type of a variable in code.**

10. Explain the difference between primitive and reference data types?

Primitive Data	Reference Data
Immutable: Their values cannot be changed.	**Mutable**: Their values can be changed.
Stored in Stack: The actual value is stored directly in the stack.	**Stored in Heap**: The reference to the value (not the actual value) is stored in the stack, while the actual object is stored in the heap.
Types: • **Number**: e.g., 42, 3.14 • **String**: e.g., "Hello" • **Boolean**: e.g., true, false • **Undefined**: A variable declared but not assigned a value. • **Null**: Represents the intentional absence of any object value. • **Symbol**: A unique and immutable primitive value.	Types: • **Object**: e.g., { key: "value" } • **Array**: e.g., [1, 2, 3] • **Function**: e.g., function() {}
Primitives are compared by value	Reference types are compared by reference.

11. How does type coercion work in JavaScript?

- **'Type coercion' is the process of converting a value from one type to another.**
  - **from String to Number**
  - **from Number to String, etc.**
- **This can happen in two ways:**
  - **Implicitly**
  - **Explicitly**

Implicit Coercion:

Here, JavaScript automatically converts types behind the scenes.

Eg;

```
let result = '5' + 3;
```

Here, JavaScript converts the number 3 to a string and concatenates it with '5'.

Explicit Coercion:

Here, manually convert a value from one type to another using built-in functions.

Eg;

```
let num = Number('42');
```

Here, the string type convert to number manually.

12. What are the typeof operator and the instanceof operator used for?

- **These operators are used to determine the type of a value and the inheritance of an object, respectively.**
  - **'typeof' Operator:**

The 'typeof' operator is used to determine the type of a given variable or value. It returns a string indicating the type of the unevaluated operand.

Syntax:

```
Typeof operand;
```

- **'instanceof' Operator:**

  The 'instanceof' operator is used to check if an object is an instance of a specific constructor or class. It returns 'true' if the object is an instance, and 'false' otherwise.

  Syntax:

  object instanceof constructor;

13. How do you convert a string to a number in JavaScript?

- **There are several ways to convert a string to a number.**
  - **'Number()' Function**
    - **The 'Number()' function converts a string to a number. If the string cannot be converted, it returns 'NaN'.**

      Eg:

      ```
      let str = "42";
      let num = Number(str);
      ```
  - **'parseInt()' Function**
    - **The 'parseInt()' function parses a string and returns an integer.**

      Eg:

      ```
      let str = "42";
      let num = parseInt(str);
      ```

      You can specify the radix (base) as the second argument to 'parseInt()'
  - **'parseFloat()' Function**
    - **The 'parseFloat()' function parses a string and returns a floating-point number. It behaves similarly to parseInt() but can handle decimal points.**

      Eg:

      ```
      let str = "42";
      let num = parseInt(str);
      ```

14. How do you convert a number to a string in JavaScript?

- **'String()' Function**
  - **The String() function converts a number to a string.**

    Eg:

    ```
    let num = 42;
    let str = String(num);
    ```

- **'toString()' Method**
  - **The toString() method, which is available on number objects, converts the number to a string.**

    Eg:

    ```
    let num = 42;
    let str = num.toString();
    ```

15. What is implicit type conversion?

- **Implicit type conversion, also known as type coercion, where the language automatically converts values from one data type to another when performing operations or comparisons.**

- **'implicit type conversion' works in many ways:**
  - **String Concatenation:**

    When a number is added to a string, JavaScript converts the number to a string and concatenates them.

  - **Numeric Operations:**

    When a string containing a numeric value is used in a mathematical operation, JavaScript converts the string to a number.

  - **Boolean Contexts:**

    When values are used in a context that expects a boolean, JavaScript converts them to boolean.

16. What are the different methods to convert a string to a number? Explain with examples.

- **The different methods:**
  - **'Number()' Function:**

```
let str = "42";

let num = Number(str);
```

  - **'parseInt() ()' Function**

```
let str = "42";

let num = parseInt(str);
```

  - **'parseFloat()' Function**

```
let str = "42.5";

let num = parseFloat(str);
```

  - **Unary Plus ('+') Operator**

```
let str = "42";

let num = +str;
```

  - **'Math.floor('), 'Math.ceil()', 'Math.round()'**

```
let str = "42.5";

let numFloor = Math.floor(parseFloat(str));

let numCeil = Math.ceil(parseFloat(str));

let numRound = Math.round(parseFloat(str));
```

  - **'parseNumber()' Function**

```
let str = "42";

let num = Number(str);
```

17. How can you handle type conversion when adding a number and a string?

- **When adding a number and a string in JavaScript, the result is typically a string due to implicit type coercion. JavaScript converts the number to a string and performs string concatenation.**

- **When add a number and a string as numbers, the string should be converted to a number before performing the addition.**

18. Explain how parseInt() and parseFloat() functions work.

- **'parseInt()' Function:**

  The 'parseInt()' function parses a string and returns an integer. It takes two arguments: the string to parse and an optional radix (base) for the numerical system.

Syntax;

parseInt(string, radix);

- **Parameters:**
  - **string: The string to be parsed.**
  - **radix: An integer between 2 and 36 that represents the base of the numeral system to be used.**
- **Behavior:**
  - **The function parses the string from left to right, stopping at the first non-numeric character.**
  - **If the radix is not specified, the default is base 10, except if the string starts with "0x" or "0X" (in which case it's parsed as a hexadecimal number).**
    - **Eg:**

      let int1 = parseInt('42'); // 42

      let int2 = parseInt('1010', 2); // 10 (binary to decimal)

- **'parseFloat()' Function:**

  The 'parseFloat()' function parses a string and returns a floating-point number. It only takes one argument: the string to parse.

Syntax;

parseFloat(string);

- **Parameters:**
  - **string: The string to be parsed.**

- **Behavior:**
  - **The function parses the string from left to right, stopping at the first non-numeric character.**
  - **It can parse decimal numbers and scientific notation.**
    - **Eg:**

```
let float1 = parseFloat('42.5');       // 42.5
```

19. What are arrays and how do you declare them?

- **Arrays in JavaScript are a type of data structure that can hold a collection of elements where those elements can be of any data type, including numbers, strings, objects, etc.**

  - **Arrays can be declare in several ways:**
    - **Using Array Literals**
      Eg: let fruits = ['apple', 'banana', 'cherry'];

    - **Using the 'Array' Constructor**
      Eg: let fruits = new Array('apple', 'banana', 'cherry');

20. What is an object in JavaScript?

An object is a complex data structure that allows you to store collections of data and more complex entities and Objects are used to represent real-world entities and are a fundamental part of JavaScript programming.