**Day1-JavaScript**

1. **Rules to declare variable names:**

* Variable names must start with a letter, an underscore (\_) or a dollar sign ($).
* Variable names cannot contain spaces.
* Variables cannot be the same as reserved keywords such as if or const.
* By convention, JavaScript variable names are written in camelCase.
* Variables should be given descriptive names that indicate their content and usage
* As JavaScript variables do not have set types, it can be useful to include an indication of the type in the name.

1. **Difference between var and let**

* Variables declared by let are only available inside the block where they’re defined.
* Variables declared by var are available throughout the function in which they’re declared.

1. **Difference between null and undefined**

Null in JavaScript means an empty value and is also a primitive type in JavaScript. The variable which has been assigned as null contains no value.

Undefined, on the other hand, means the variable has been declared, but its value has not been assigned.

1. **What are the other data types available in JS?**

* String
* Number
* Bigint
* Boolean
* Undefined
* Null
* Symbol
* Object

The object data type can contain:

* An object
* An array
* A date

1. **JS Components**

Components are independent and reusable bits of code. They serve the same purpose as JavaScript functions, but work in isolation and return HTML.

Components come in two types, Class components and Function components.

1. **How a JS program executes**

* When the JavaScript engine scans a script file, it makes an environment called the **Execution Context**that handles the entire transformation and execution of the code.
* During the context runtime, the parser parses the source code and allocates memory for the variables and functions. The source code is generated and gets executed.
* There are two types of execution contexts: **global** and **function**. The global execution context is created when a JavaScript script first starts to run, and it represents the global scope in JavaScript. A function execution context is created whenever a function is called, representing the function's local scope.

1. **Operator Precedence**

* Operator precedence describes the order in which operations are performed in an arithmetic expression.
* Multiplication (\*) and division (/) have higher **precedence** than addition (+) and subtraction (-).

1. **List of Operators**

* Arithmetic Operators
* Assignment Operators
* Comparison Operators
* String Operators
* Logical Operators
* Bitwise Operators
* Ternary Operators
* Type Operators