1. **Explain various types of console functions?**

* **console.log()**

Mainly used to log(print) the output to the console. We can put any type inside the log(), be it a string, array, object, boolean etc.

* **console.error()**

Used to log error message to the console. Useful in testing of code. By default the error message will be highlighted with red color.

* **console.warn()**

Used to log warning message to the console. By default the warning message will be highlighted with yellow color

* **console.count()**

This method is used to count the number that the function hit by this counting method.

1. **Difference between VAR , LET and Const with examples?**

* var declarations are globally scoped or function scoped while let and const are block scoped.
* var variables can be updated and re-declared within its scope; let variables can be updated but not re-declared; const variables can neither be updated nor re-declared.
* They are all hoisted to the top of their scope. But while var variables are initialized with undefined, let and const variables are not initialized.
* While var and let can be declared without being initialized, const must be initialized during declaration.
* EXAMPLE:

{

const PI=3.14;

console.log(PI);

} //block 1

{

console.log(PI);

} //block 2

/\* Since we are using "const PI=3.14", scope of "PI" is limited to block 1 and "PI" is not recognized in block 2 \*/

function block1() {

var a=10;

console.log(a);

} //function scope of block 1

function block2() {

a++;

console.log(a);

} //function scope of block 2

/\* Since we have enclosed block1 and block2, within separate functions, the scope of "var a=10", is limited to block 1 and "a" is not recognized in block 2 \*/

{

let a=10;

console.log(a);

} //block 1

{

a++;

console.log(a);

} //block 2

/\* Since we are using "let a=10", scope of "a" is limited to block 1 and "a" is not recognized in block 2 \*/

1. **Brief Intro on Types of Datatypes in JavaScript?**

**The String Data Type**

The string data type is used to represent textual data (i.e. sequences of characters). Strings are created using single or double quotes surrounding one or more characters.

**The Number Data Type**

The number data type is used to represent positive or negative numbers with or without decimal place, or numbers written using exponential notation e.g. 1.5e-4 (equivalent to 1.5x10-4).

**The Boolean Data Type**

The Boolean data type can hold only two values: true or false. It is typically used to store values like yes (true) or no (false), on (true) or off (false), etc. as demonstrated below:

**The Undefined Data Type**

The undefined data type can only have one value-the special value undefined. If a variable has been declared, but has not been assigned a value, has the value undefined.

**The Null Data Type**

This is another special data type that can have only one value-the null value. A null value means that there is no value. It is not equivalent to an empty string ("") or 0, it is simply nothing.

**The Array Data Type**

An array is a type of object used for storing multiple values in single variable. Each value (also called an element) in an array has a numeric position, known as its index, and it may contain data of any data type-numbers, strings, booleans, functions, objects, and even other arrays. The array index starts from 0, so that the first array element is arr[0] not arr[1].

**The Function Data Type**

The function is callable object that executes a block of code. Since functions are objects, so it is possible to assign them to variables.

**The typeof Operator**

The typeof operator can be used to find out what type of data a variable or operand contains. It can be used with or without parentheses (typeof(x) or typeof x).