Javascript is a object oriented, object based scripting language,which is used to develope functional part of the web page.

Characterstics of javascript:-

-> It is a programming language or scripting language, which means it will create a response with respect to the program.

-> It is a dynamic language which means the javascript code can be updated in upcomming lines.

-> It is a Synchronous language, which means the browser engine executes the programm line by line.

-> It is a single threaded language, the javascript have single thread or single stack which is call stack.

-> It is a high level language which means javascript is human understandable language

-> It is a interpreted language

Ways to write Javascript program:-

-> Javascript program can be written/performed in two ways they are

- Internal Javascript

- External javascript

.Internal Javascript:-

.It is performed by using Script tag which we have to specify inside the body tag after writing Html part of the program.

Eg:

<body>

<h1 id="heading"></h1>

<script>

document.getElementById("heading").innerHTML="This is inside the H1 which is on UI"

</script>

</body>

.External javascript:

-It is performed by using external javascript document.

-We have to link the external javascript by using script tag.

-In the script tag we have to specify src attribute, for the src attribute we have to pass url of external javascript document.

-We use differ attribute to specify the browser to run Html document at first then javascript document

Eg:- HTML document:-

<head>

<title>Document</title>

<script src="externaljs.js" defer async></script>

</head>

<body>

<h1 id="heading">Hello</h1>

</body>

</html>

Javascript document :- document.getElementById("heading").innerHTML = "This is external js";

Javascript Printing Statements:-

-In js we have printing statements like:

console.log(), document.write(), document.writeln(), window.alert, innerHTML, innerText.

.console.log():

-It is used to debug the code.

-The console.log() prints the output in browser console.

-In console.log(), console is a object log() is a function which is inside the console object and it is accessed by using "." notation.

Eg:

console.log("This is in browser console");

console.log(10 + 5);

console.log(console);

.document.write():-

- It is used to print the output on UI.

-In document.write(), document is a object which refers to current html document, write() is a function which is inside the document object

and it is accessed by using "." notation.

-We don't use document.write() on real time projects because it have its disadvantages like:

->It will replace the parent document.

->The document.write will not work if the program is asynchronously imported.

Eg: document.write("This is on <b style='color:red'><i> User</i></b> interface");

.document.writeln():-

-It is similar to document.write, the difference between document.write() and writeln() is, it gives single space between the statements.

Eg:

document.writeln("<br> One");

document.writeln("Two");

.window.alert():-

-It is used to print the output on dialog of browser or pop-up window of the browser.

-The dialog will wait until the user dismisses the dialog box by clicking on ok button.

-In window.alert(), window is a object alert is a function which is inside the window object, to access the alert we have to use '.' notation.

-Whenever a function or a object or a property which have direct relationship with the window object there is no need to specify the window object.

Eg:

window.alert("OOPS");

alert("this is second alert");

.innerHTML:

-It is a property which is used to ask the value inside the html elements.

-To use innerHTML property we have to use DOM methods to target the element.

-In innerHTML it will consider html tag as a tag.

Eg:

HTML document:

<h1 id="heading"></h1>

javascript document:

document.getElementById("heading").innerHTML = "This is inside H1 tag";

.innerText:-

-It is used to pass the value inside the html elements.

-To use the innerText we have to use DOM methods to target the element.

-In innerText if we specify html elements it will be consider as text.

Eg:

Html document

<span id="heading1"></span>

Javascript document

document.getElementById("heading2").innerText = "<p>content int js</p>";

Javascript Tokens:-

-Token is a smallest unit of programming language.

-In javascript we have tokens like:

.Variable

.Identifier

.Value

.Variable:

-Variable are used to allocate the memory block, in javascript we have 3 types of Variable they are

.Var

.let

.const

.Var:

-var is a functionscope variable in which we can performe hoisting, in var we can re-declare and re-initialize the variable.

Eg:

var num=10;

console.log(num);

.Let:

-Let is a variable with block scope, In let we can perform re-initialization but we cannot perform re-declaration and hoisting.

Eg:

let num=20;

console.log(num);

.Const:

-Const is a Block scope variable, if we declare the const in the same statement we have to initialize, the const value must kept constant throughout

the program which means we cannot re-initialize and re-declare.

-In const we cannot perform variable hoisting

Eg:

const num=20;

console.log(num);

.Variable hoisting:

-using the variable before its declaration is called hoisting.

-In javascript we can perform variable hoisting only in var

Eg:

console.log(a);

var a=10;

console.log(a);

.Difference between Var,Let and Const

Status | Var | Let | Const

--------------------------------------------------------|-----------------|-------------|-----------------------------------------------

Declaration & initialization | Y | Y | Y

Only declaration | Y | Y | N

Declaration after initialization | Y | Y | Y

Re-declaration | Y | N | N

Re-initialization | Y | Y | N

Scope | FUNCTION | Block | BLOCK

Hoisting | Y | N | N

.Identifier:

- Identifier are the names given for the memory block to write Identifier we have to follow some rules

.We have to write Identifier with camel casing.

.We should not start Identifier with numbers or special Characters except $ and \_ .

.We should not use keywords as identifier.

-We should not give space in between the identifier.

-There is no limit for length of identifier but we may face some problem with some javascript engines if the identifier is longer than 31 characters.

.Value:

-Values are nothing but data which is stored in the memory block.

-In javascript we have two types of data type they are:

.Primitive data type.

.Non-Primitive data type.

.Primitive :

-In javascript in Primitive data type it will not return object type and it represents single value and immutable.

-In javascript we have Primitive data types like:

.Number - Eg: var num=10;

.String - Eg: car str="Ayush";

.Undefined-Eg: var un=undefined;

.Null - Eg: var nu=null;

.Symbol - Eg: var sym=Symbol();

.bigint - Eg: var num2 = BigInt(12365468497465125468965132549);

.Boolean - Eg: var bool=true;

\*\*NOTE\*\* - NULL IS A PRIMITIVE DATA TYPE EVEN AFTER THE RETURN TYPE IS OBJECT,BECAUSE IT RETURNS EMPTY OBJECT.

.Non-Primitive data type:

-Non Primitive data types are also called as object type because the typeOf of Non-Primitive data type is object type.

-In javascript we have non-primitive data types like:

.function

.Array

.Object

.typeOf:

-It is a operator which returns the data type of javascript values.

-It always return string type output.

Eg:

console.log(typeOf "Ayush");

console.log(typeOf 'a');

console.log(typeOf true);

Typecasting:

-It is also known as type conversion, it is a process where we covert one datatype to another datatype.

-In typecasting we have two types they are:

.Implicit type conversion.

.Explicit type conversion.

.Implicit type conversion:

-In Implicit type conversion the data is going to convert automatically without any external function or constructor.

Eg:

\*\*\*\*\*\*CONVERSION TO STRING TYPE:\*\*\*\*\*\*\*

var a = 10;

console.log(typeof (a), a);

var b = "10";

console.log(typeof(b),b)

\*\*\*\*\*\*CONVERSION TO NUMBER TYPE:\*\*\*\*\*\*\*

var a="10";

var b=10;

var c=a\*b

console.log(typeOf c,c)

\*\*\*\*\*\*CONVERSION TO BOOLEAN TYPE:\*\*\*\*\*\*\*

var a=10;

var b=Boolean(a)

console.log(typeOf (a==b),a==b)

.Explicit type conversion:

-it is called as manual type conversion in whichwe use function and constructor

.conversion to Number type:

-To convert the data into number type we use:

.Number()

.ParseInt()

.ParseFloat()

Eg:

var a = "10.5a";

console.log(typeof a,a);

var a=Number(a)

console.log(typeof(a),a)

var a = parseInt(a);

console.log(typeof a,a)

var num = parseFloat(a);

console.log(typeof num,num);

\*\*NOTE\*\* - Whenever we try to convert other datatypeinto number type if it fails it will return NaN, Which stands for Not a Number which is a special type of number.

.Conversion to string type:

-To convert other data into string type we can use:

.String()

.toString()

Eg:

var num = 10;

console.log(typeof num, num);

var str = String(num);

var str = num.toString();

console.log(typeof str, str);

.Conversion to Boolean type:

-TO convert the data into Boolean type we use Boolean constructor.

Eg:

var vari=null;

console.log(typeOf vari,vari);

var bool=Boolean(vari);

console.log(typeOf bool,bool);

Javascript Input Statements:

-In javascript we have two Input statements they are:

.confirm();

.prompt();

.confirm():

-It is a function which is used to collect the boolean value from the dialog box which returns "true" or "false".

Eg:

var bool: window.confirm("Let's party tonight!");

console.log(bool);

.prompt():

-It is used to collect the String type input from the user through the dialog box.

Eg:

var userName = prompt("What is your name");

console.log(userName);

Javascript operator:

-Operators are used to perform the operations between the operands.

-In javascript we have operators like:

.Arithmetic operator

.Assignment operator

.Comparison operator

.Logical operator

.Bitwise operator

.Increment and Cecrement

.Arithmetic operations:

-It is used to perform Arithmetic operation between the operands.

-In arithmetic operator we have operators like:

.Addition(+)

.Substraction(-)

.Multiplication(\*)

-Division(/)

-Modulus(%)

-Exponentation(\*\*)

Eg:

var num1 = 10;

var num2 = 2;

console.log(num1+num2);

console.log(num1-num2);

console.log(num1\*num2);

console.log(num1/num2);

console.log(num1%num2);

console.log(num1\*\*num2);

.Assignment operations:

-It is used to assign the value to javascript variables.

-In assignment we have =,+=,-=,\*=,/=,%= and \*\*=.

Eg:

var num1 = 10;

var num2 = 2;

num1 += num2;

num1 -= num2;

num1 \*= num2;

num1 /= num2;

num1 %= num2;

num1 \*\*= num2;

console.log(num1, num2);

.Comparison operators:

-It is used in logical statements to determine equality or difference between variables or values.

-In comparison operators we have operators like:

.== : First operand value second operand value.

.=== : First operand value and data type equals to the second operand value and data type.

.!= : First operand value not equal to the second operand value.

.!== : First operand value and data type not equal to the second operand value and data type.

.> : First operand value greater than second operand value.

.>= : First operand value greater than or equal to second operand value

.< : First operand value less than second operand value.

.<= : First operand value less than or equal to second operand value.

Eg:

var a = 10;

var b = 10;

console.log(a==b);

console.log(a===b);

console.log(a!=b);

console.log(a!==b);

console.log(a>b);

console.log(a>=b);

console.log(a<=b);

console.log(a<=b);

console.log(a==b==c);

.Logical operator:

-It is used to perform the Logical operations in the statements.

-It returns true or false.

-In javascript we have logical operators like:

.AND (&&)

.OR (||)

.NOT (!)

Eg:

var a = 10;

var b = 10;

var c = 10;

console.log(a == b && b == c);

console.log(!true);

.Bitwise operators:

-In Bitwise operators we have

.&, |, ~, >>, <<, >>>

Eg:

console.log(a==b&b==c);

console.log(a==b|b==c);

.Increment And Decrement:

-To increment the variable with 1 we use ++.

-To decrement the variable with 1 we use --.

Eg:

var a=1;

console.log(a++);

console.log(a);

console.log(++a);

console.log(a--);

console.log(--a);