**Variable** are the containers, used to store the values.

**Data type**(to be used in javascript)-

1. number e.g. var a = 10
2. string e.g. var str = ”Hello”
3. boolean e.g. var decision = true;
4. undefined e.g. var b;
5. null e.g. var c = null
6. symbol

To store these data types we will use **var.**

e.g. var a=10

If we put ; or not in java script it will not give an error

Declare and defined: When we assign a value to a variable, then it is defined e.g. var a = 10

**Types of Datatypes:-**

1. Primitive Data Types- {number, string, Boolean, null, undefined, symbol}
2. Referenced Data Types – {Array, Object}

**Array** are collection of elements(Any type of value). E.g. var arr=[2,3,4,5,9 ]

**Reference** is a particular position given to a element in array.

**Object** data type in java script is a key, value pair. For e.g. var marks ={Ram:35; Sham:76; Raman:80;} {key:value}

**The Operators in java script:**

1. Arithmetic Operators (+,-,\*,/,//)
2. Assignment operators (=):- Assign or copy the value from right hand side to left hand side variable.
3. Equality/comparison Operators (==,<=,>=,>,<,!=)
4. Logical Operators (AND, OR ,NOT)

**Functions in Java Script:** function is a

e.g.:-

function avg(a,b)

{

Var c = (a+b)/2

Return c;

}

**Type of Functions:**

1. With argument without return -
2. With argument with return value -
3. Without argument with return value -
4. Without argument without return value –

**Conditional Statement in Java Script**:(if, else)

**Array Operations Methods -**

1. arr.pop() - Pop method delete the last element for the given array.
2. arr.push(“ram”) - Used to insert new element at the end of the given array.
3. Arr.shift() - It is used to remove the first element from the array.
4. Arr.unshift(“xyz”) -

**String operations method –**

1. Length
2. indexOf(“string”)
3. lastIndexOf(“string”)
4. slice(startposition, character\_till\_position)
5. replace(“string-old”,”string-new”)

**Normal –**

1. function isPositive (number)

{

return (number>0)

}

**Arrow Function –** In arrow function we need not to write return keyword explicitly.

Steps:

1. let isPositive = function(n)

{

return(n>0)

}

1. let isPositive=(n) => {

return(n>0)

}

1. let isPositive = (n) => n>0
2. let isPositive = n => n>0

Zero : let msg = () => “Hello”

Single : let isPositive= (n) => n>0,

or

let isPositive = n => n>0,

Multiple: let sum = (a,b) => a+b,

**Declaring using function expression –**

var sum = function(a,b)//using fuction expression

{

res=a+b

console.log(res);

}

Note: TypeError, because it host “var sum = undefined ” at top.

\*\*function using expression and arrow function do not support hosting

var sum = function(a,b)

{

res = a+b;

console.log(res);

}

console.log(sum);

console.log(sum);

var sum = function(a,b)

{

res = a+b;

console.log(res);

}

**Object –**

var student = {

fname : “Ram”,

lname : “Kumar”

class : “B.Tech-CSE”

disp : function()

{

console.log(fname);

console.log(lname);

console.log(class);

}

}

**This keyword -** this keyword refers to the object, it belongs to it has different values depending on where it is used

1. alone, this refers to the global object
2. In a regular function this refers to the global object

<script>

function add()

{

var a = 10;

var b = 20;

var c = a+b;

console.log(this);

console.log(c);

}

add();

1. In a method this refers to the owner object

<script>

var student ={name: “Ram”,

class: “BE”,

add: function()

{var a = 10;

var b = 20;

var c = a+b;

console.log(this);

console.log(c);

}

add();

1. In a function in strict mode this keyword will show as undefined

example

function student(first, last, age, eye){

this.firstName = first;

this.lastName = last;

this.age = age;

this.eyecolor = eye;

this.dispName = function () {

return this.firstName + “ ” + this.lastName;

}

}

const aman = new student(“Aman”, “Singh”, 20, “Black”);

const suman = new student(“Suman”, “Garg”, 22, “Brown”);

const rajiv = new student(“Rajiv”, “Verma”, 18, “Blue”);

console.log(aman.age);

console.log(rajiv.age);

console.log(aman.dispName);

console.log(suman.dispName);

**Difference between events and method:-**

events are always fired

methods need to be called to execute there task

events are active entities

methods are passive entities

AJAX stands for As