(IIFE)immediately invoked function expression

```
(function functionName() {
  console.log("Hello World");
})();
```

```
(function functionName() {
  console.log("Hello World");
})();
```

local and outer variables

```
let userName = 'John';

function showMessage() {
  let message = 'Hello, ' + userName;
  alert(message);
}

showMessage(); // Hello, John
```

Array Items – some method

```
let marks = [10,60,80,40]
    let result = marks.some((value)=>{
      return value > 30
    })
    console.log(result)
```

Array Items – every method

```
let marks = [10,60,80,40]
	let result = marks.every((value)=>{
	   return value > 30
	})
	console.log(result)
```

Set (stores unique value)

```
let s = new Set(['apple','mango','orange','apple','apple’])
     alert(s.size)
     s.add('cherry’)
     alert(s.size)
     alert(s.has('apple’))
     s.delete(‘apple’)
console.log(s[0])
   //   s.clear();
   //   console.log(s.size);
     s.forEach((value,valueagain,set)=>{
       console.log(value,valueagain,set)
     })
```

Use strict mode

-
  ```
  <script>
   "use strict";  // try without strict mode
   x = 34; // let x = 34;
   console.log(x);
   delete x; //deleting variable or function not allowed
   console.log(x);
   function add(x,x){ //duplicate argument not allowed
       console.log(x+x)
   }
   add(3,7)
  </script>
  ```

data-* custom attribute

The data-* is a custom attribute that allow us to embed custom attributes on all HTML elements. It can be used in JavaScript. We can also use these data attributes to apply styles.

……………………………………………………………….

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      div[data-type="car"] {
        background-color: yellow;
      }
    </style>
  </head>
  <body>
    <div id="v1" data-type="car" data-wheels="4">Car</div>
    <div id="v2" data-type="bike" data-wheels="2">Bike</div>
    <script>
      console.log(v1.dataset.type);
    </script>
  </body>
</html>
```

Table Elements - insertrow

```
<input type="text" id="t1" placeholder="Enter Name" /><br /><br />
<input type="text" id="t2" placeholder="Enter Name" /><br /><br />
<input type="button" onclick="myFunction()" value="Submit" />
<table id="t" border="1"></table>
  <script>
    function myFunction() {
      var row = t.insertRow(0);  // remove 0 to insert in the end
      var cell1 = row.insertCell(0);
      var cell2 = row.insertCell(1);
      cell1.innerHTML = t1.value;
      cell2.innerHTML = t2.value;
      document.body.append(t);
    }
  </script>
```

Navigating Table Elements

```
<html>
  <body>
    <table border="1" id="mytable">
      <caption>Marks Sheet</caption>
      <tr><th>Subject</th><th>Marks</th></tr>
      <tr><td>Science</td><td>90</td></tr>
    </table>
    <script>
      console.log(mytable.caption);
      for (let i=0;i<mytable.rows.length;i++){
        let s = mytable.rows[i].cells[1]
        if (s.innerHTML > 70) s.style.backgroundColor="green“
      }
    </script>
  </body>
</html>
```

Data Types (Primitive/Value type)

1. let n=2;
2. let s = “Hello World”; //double or single quote
3. let flag = true; //true or false - boolean
4. let name; //undefined
5. let cost=null;

Data Types (Reference Type)

1. Objects
2. Arrays
3. Functions

toLocaleUpperCase

```
let str = "India"
        console.log(str.toLocaleUpperCase('en-US'))
        console.log(str.toLocaleUpperCase('TR')) //turkish
```

Chanining methods

```
 let name="john"
        name = name.charAt(0).toUpperCase()+name.slice(1)
        console.log(name)
```

……………….chanining methods in nodejs

Chanining methods

```
<script>
   let calc = function () {
    this.i = 0;
    this.add = function (i) {
     this.i = this.i + i;
     return this;
    };
    this.subtract = function (i) {
     this.i = this.i - i;
     return this;
    };
    this.print = function () {
     console.log(this.i);
    };
   };
   let c = new calc();
   c.add(20).subtract(5).print();
  </script>
```