```javascript
// number methods
// toString()

const number = 12354672456127596

console.log(typeof number)

const result = number.toString()

console.log(typeof result)


// // toFixed()

// const num1 = 5.0872346652

// const result = num1.toFixed(5)
// console.log(result)

// // to fixed method is used to run up a to a defined decimal place



// valueOf()

let num1;
console.log(num1.valueOf())

function (num1) {
    return num1.valueOf()
}




// arrays

// An array is a special variable, which can hold more than one value

const car = ["toyota", "benz"]
const List = ["lagos", "ibadan", "ogun", "Ekiti", "osun" ]

// const alpha = "abcdefghijk"

console.log(List[3])



// array methods

// Array length

// const List = ["lagos", "ibadan", "ogun", "Ekiti", "osun", "edo", "anambra", "imo", "kogi"]
```

```javascript
// const result = List.length
// console.log(result)


// Array toString()

// const List = ["lagos", "ibadan", "ogun", "Ekiti", "osun", "edo", "anambra", "imo", "kogi"]
// const result = List.toString()
// console.log(typeof result)

// Array pop()

// const List = ["lagos", "ibadan", "ogun", "Ekiti", "osun", "edo", "anambra", "imo", "kogi"]
// // const result = List.pop()
// console.log(List)

// Array push()

// use let instead of const

// let List = ["lagos", "ibadan", "ogun", "Ekiti", "osun", "edo", "anambra", "imo", "kogi"]
// List.push("Abuja")

// console.log(List)

// Array shift()

// const List = ["lagos", "ibadan", "ogun", "Ekiti", "osun", "edo", "anambra", "imo", "kogi"]
// List.shift()

// console.log(List)

// Array unshift()

// use let instead of const
// let List = ["lagos", "ibadan", "ogun", "Ekiti", "osun", "edo", "anambra", "imo", "kogi"]
// List.unshift("Abuja")

// console.log(List)

// Array join()

// use Let instead of const
// let Cars1 = ["lambo", "bentley", "rolls", "martins"]
// // const cars2 = ["rolls ", "martins "]
// const result = Cars1.join(",")
// console.log(result)




// Array concat()

// use let instead of const
// let Cars1 = ["lambo", "bentley", "rolls", "martins"]
// const cars2 = ["rolls ", "martins"]
// const result = Cars1.concat(cars2)
// console.log(result)
```

```javascript
// Array Slice()

// const List = [ 'lambo', 'bentley', 'rolls', 'martins', 'rolls ', 'martins ' ]
// // List.slice(1, 4)
// const result = List.slice(1, 4)
// // console.log(result)
// console.log(List)

// Array splice()

const fruits = ["Banana", "Orange", "Apple", "Mango"];

fruits.splice(2, 0, "coconut", "paw paw")

console.log(fruits)

// The first parameter (2) defines the position where new elements should be added (spliced in).
// The second parameter (0) defines how many elements should be removed.
// The rest of the parameters ("Lemon" , "Kiwi") define the new elements to be added.

// The splice() method returns an array with the deleted items:


// Array concat()

// use let instead of const
// let Cars1 = ["lambo", "bentley", "rolls", "martins"]
// const cars2 = ["rolls ", "martins"]
// const result = Cars1.concat(cars2)
// console.log(result)

// Array Slice()

// const List = [ 'lambo', 'bentley', 'rolls', 'martins', 'rolls ', 'martins ' ]
// // List.slice(1, 4)
// const result = List.slice(1, 4)
// // console.log(result)
// console.log(List)

// Array splice()

const fruits = ["Banana", "Orange", "Apple", "Mango"];

fruits.splice(2, 0, "coconut", "paw paw")

console.log(fruits)

// The first parameter (2) defines the position where new elements should be added (spliced in).
// The second parameter (0) defines how many elements should be removed.
// The rest of the parameters ("Lemon" , "Kiwi") define the new elements to be added.

// The splice() method returns an array with the deleted items:


// maths
// Math.round(x)   Returns x rounded to its nearest integer
```

```javascript
// console.log(Math.round(5.6))

// Math.ceil(x)       Returns x rounded up to its nearest integer

// console.log(Math.ceil(5.124827350896723895672398057609082))

// Math.floor(x)      Returns x rounded down to its nearest integer

// console.log(Math.floor(5.6))
// Math.trunc(x)
// console.log(Math.trunc(5.824827350896723895672398057609082))



// comparison
// condition
```