Important Object Methods And Immutability

```
let product = {
        name : 'IPhone 14',
        company: 'Apple',
        quantity: 5,
        price: 125000
        waranty: "2 years"
}
```

So this is an object and an object can have a lot of key and value pairs but in an object the key cannot be duplicate so it has to be unique but the values can be duplicate.

Object.keys()

if we want to access all the keys from an object then we can use a inbuilt function `Object.keys(objectname)`

This method takes an argument and the argument is the object name itself
in this example my object name is **product**

```
let product = {
        name : 'IPhone 14',
        company: 'Apple',
        quantity: 5,
        price: 125000
        waranty: "2 years"
}

// keys:
console.log(Object.keys(product))
```

it returns an array of strings and each array represent an unique key.

```
['name', 'company', 'quantity', 'price', 'waranty' ]
```

Object.values()

The get all the values from an object we have to use `Object.values(objectname)`
This will also give us an array of strings.

```
console.log(Object.values(product))
['name', 'company', 'quantity', 'price', 'waranty' ]
```

Object.entries()

There is an inbuilt function which can give us both values and key from an object ==**`Object.entries(objectname)

This gives us an array of array where 0th index is for key and the 1st index is for value like the other two methods this one will also take one argument and the argument is the object name

```
console.log(Object.entries(product))
[
  [ 'name', 'IPhone 14' ],
  [ 'company', 'Apple' ],
  [ 'quantity', 5 ],
  [ 'price', 125000 ],
  [ 'waranty', '2 years' ]
]
```

how can we check the number of key value pairs in an object.?

We have to find the length of an key and value pairs otherwise just get the keys or value using `Object.keys(product) or Object.values(product)` and then find the length

```
console.log(Object.entries(product).length)
console.log(Object.keys(product).length)
console.log(Object.values(product).length)
```

we can use any one of them we get the same result.

Mutability and Immutability in JS Objects

So in every programming languages we will get to see mutability and immutability concepts

mutable mean we can do changes it allow us to do the changes lets say for an example we created a variable and initialize a value to it and later on if we want to change the value of the same variable then this is called mutable.

immutability mean where we cannot do the modifications, if you want to make an variable/object as immutable then instead of using let and `var` just use `const` because `const` is something which doesn't allow us to change

it actually behave differently with variable and with objects lets get into it with an example. and one more thing if we are defining with the `const` then we have to initialize some value into it we just cannot keep it empty

```
const userName; not allowed
SyntaxError: Missing initializer in const declaration`
```

we need to initialize some value into it otherwise we will get an error as

```
const userName = "deepak"
userName = "rahul" // this is not allowed
TypeError: Assignment to constant variable.

const userName = "netra"
SyntaxError: Identifier 'userName' has already been declared
```

so from the examples it is very clear that anything we declared with `const` we cannot update it we cannot do any changes to it.

but it will behave different with object lets create an object with `const` and try to do some changes into it

```
const product = {
        name : 'IPhone 14',
        company: 'Apple',
        quantity: 5,
        price: 125000
        waranty: "2 years"
}
product.name = "Samsung 15"
product.company = "Samsung"

console.log(product)

{
  name: 'Samsung 15',
  company: 'Samsung',
  quantity: 5,
  price: 125000,
```

```
  waranty: '2 years'
}
```

see the changes are getting reflected that's' what I am talking about
`const` doesn't allow the reassignment if I will try to reassign then I will get same error
but here I am not doing any reassignment, I am just changing the value which are inside the object if I want to create a new object altogether with same object name then this is not allowed for example

```
const product = {
        name : 'IPhone 14',
        company: 'Apple',
        quantity: 5,
        price: 125000
        waranty: "2 years"
}

product = {
        name: 'Samsung 15',
        company: 'Samsung',
        quantity: 5,
        price: 125000,
        waranty: '2 years'
}
```

now lets go more deeper there is something called heap and stack, Objects are stored in the heap memory and if we are defining a variable and inside the variable lets say we are saving some number or string that will get store inside the stack

so ham jo bhi objects banate hein woh object jake store hota hein Heap memory mein / or woh object jake banta hain Heap Memory Region mein, Object ko banane ke liye ham jo variable use karte hein woh jake store hota hein stack mein

Why do we need immutability

lets say for an example there is a DB Connection and there are very less cases where we want to update the configuration of DB Connection, Most of the time once the connection is established there is no need of changing it and this is a good example of immutability