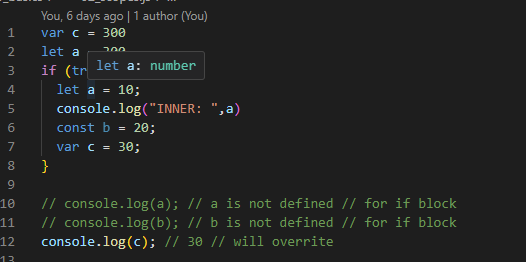
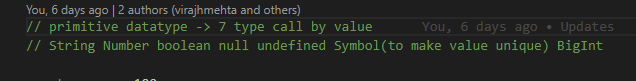
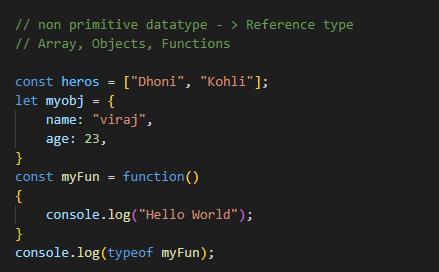
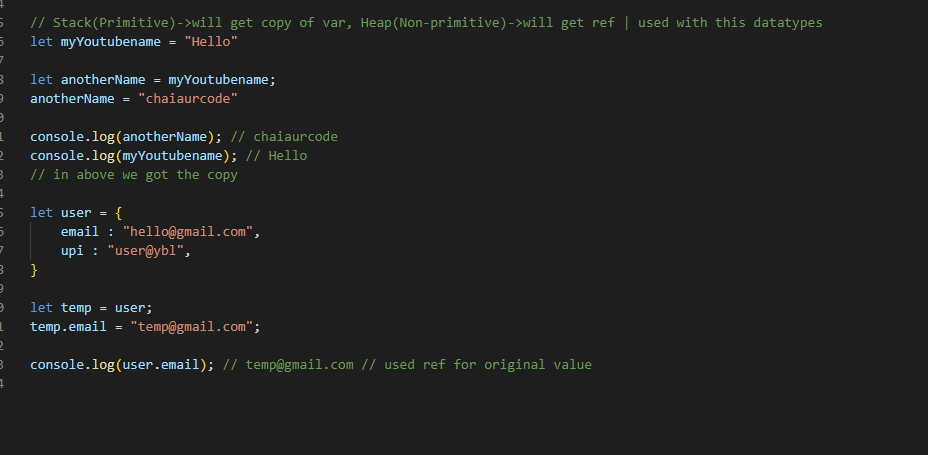
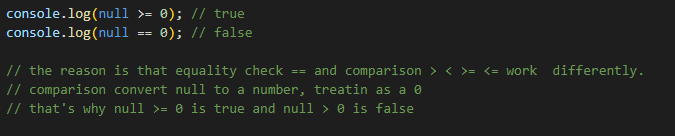
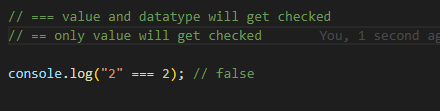
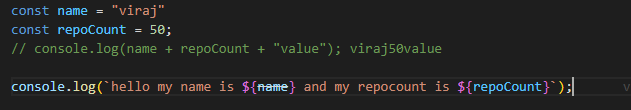
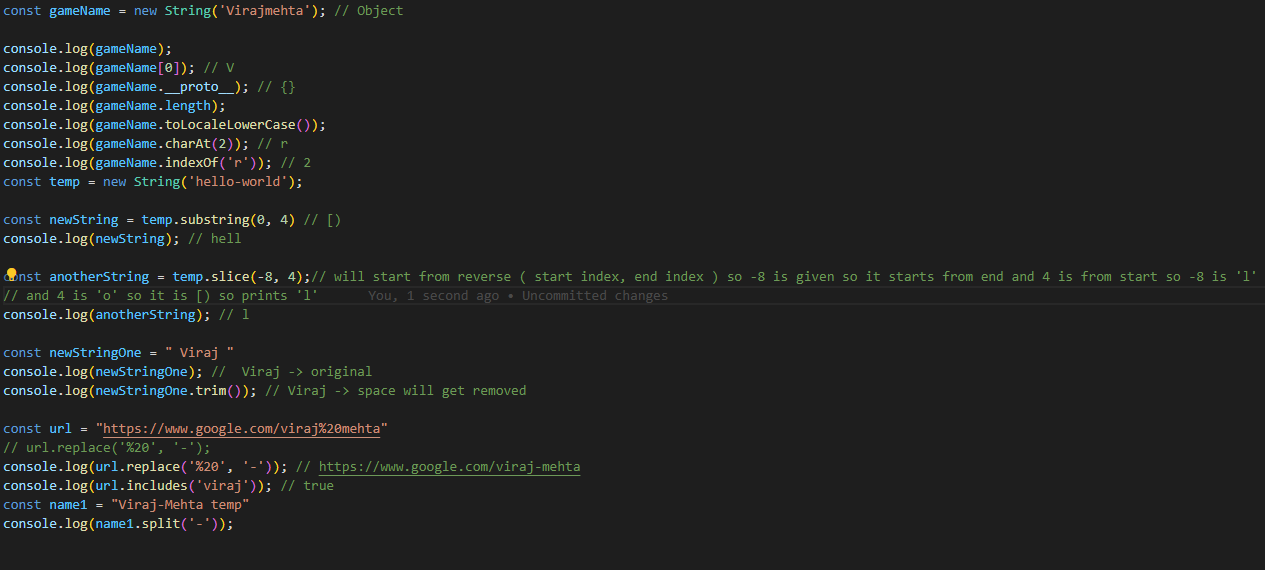
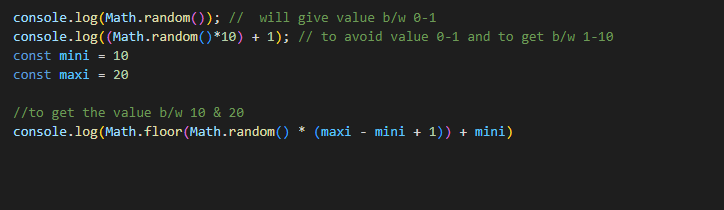
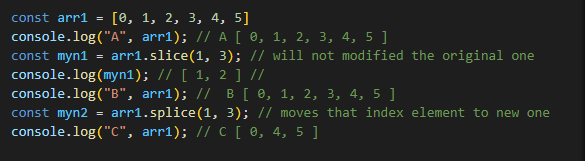
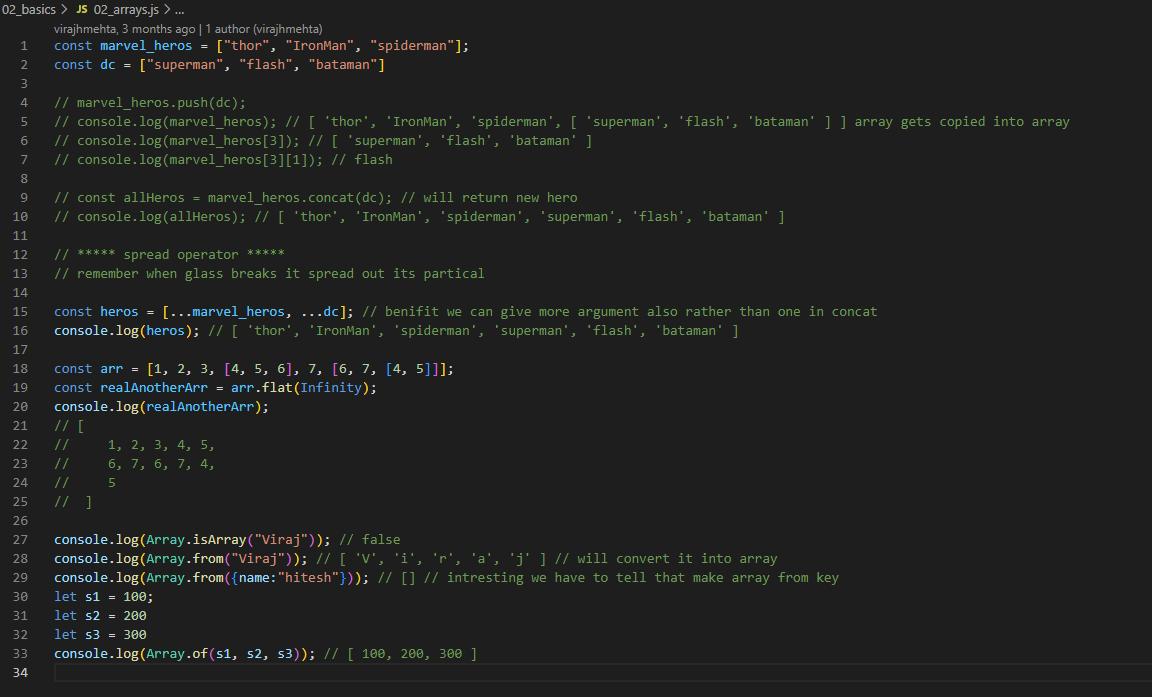
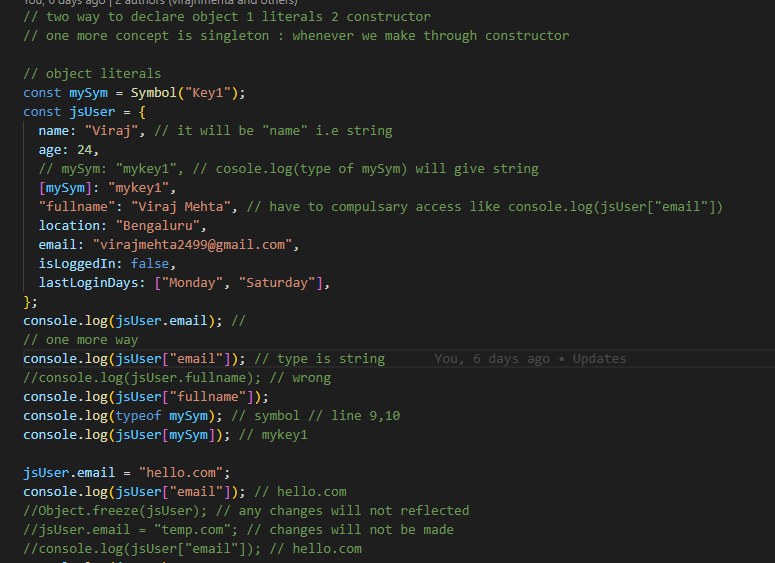
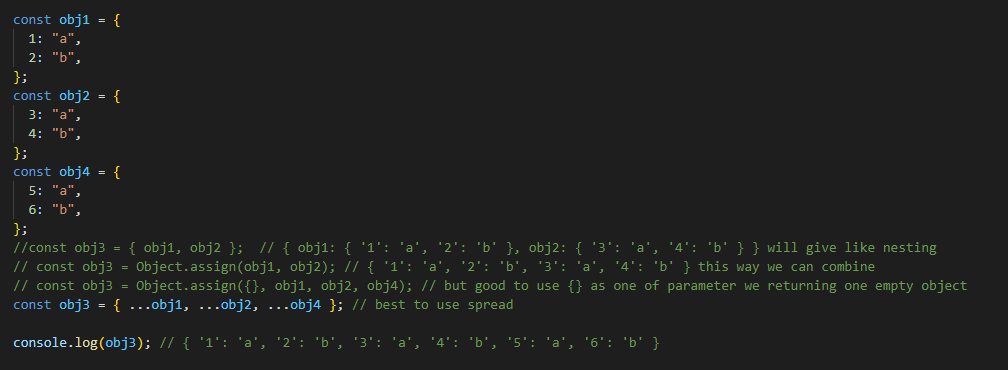
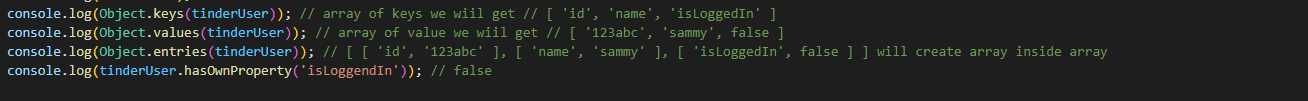
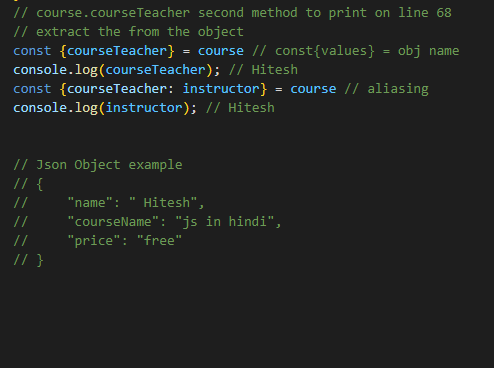
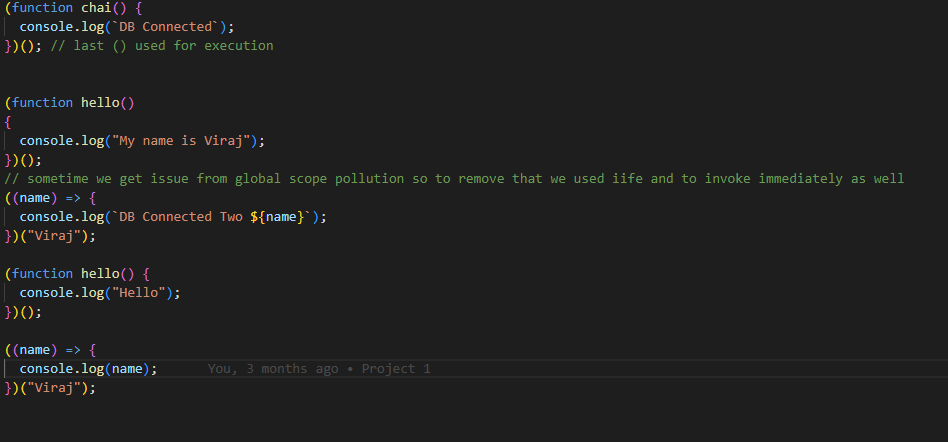
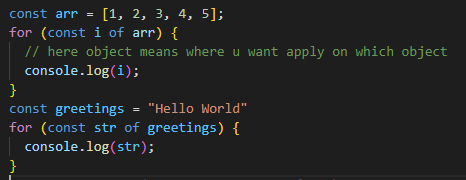
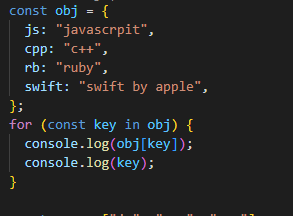
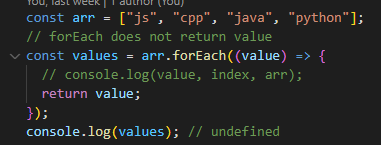
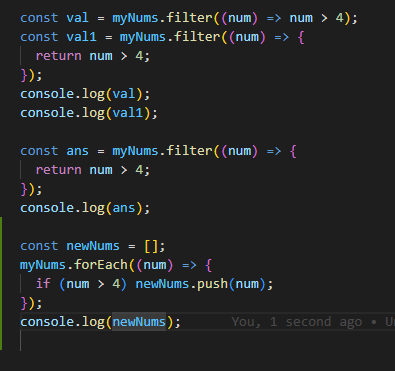
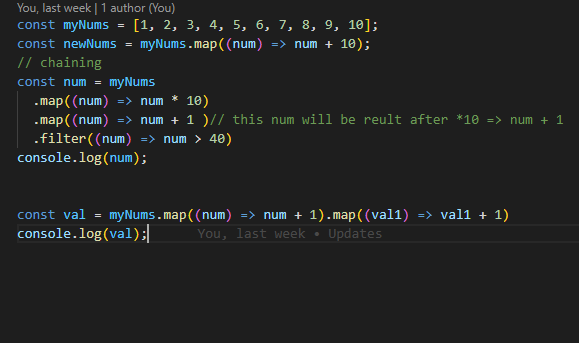
Notes:

1. Variables:
   1. Const : The **const** keyword has all the properties that are the same as the let keyword, except the user cannot update it and have to assign it with a value at the time of declaration. These variables also have the **block scope**. It is mainly used to create **constant** variables whose values can not be changed once they are initialized with a value.
   2. let: The **let** keyword is an improved version of the var keyword. It is introduced in the ES6 or EcmaScript 2015. These variables has the **block scope**. It can’t be accessible outside the particular code block ({block}).
   3. Var: The **var** is the oldest keyword to declare a variable in JavaScript. It has the **Global scoped or function scoped** which means variables defined outside the function can be accessed globally, and variables defined inside a particular function can be accessed within the function.
   4. If we are **just writing** the variable name then it will consider as “var” only
   5. 

let accountState; //default value undefined

* + 1. Default value will be undefined

1. DataTypes:
   1. Primitive:
      1. 
      2. typeof null -> object
      3. typeof undefined -> undefined
      4. Symobl -> assigns unique value
   2. Non primitive
      1. 
   3. 
2. DataType Conversion
   1. 
   2. null vs undefined : null is an assigned value. It means nothing. undefined means a variable has been declared but not defined yet. null is an object.
   3. Null default value is 0.
      1. 
   4. 
   5. 
3. String
   1. We can either declare with ‘ ‘ or “ “
   2. Other Way: 
   3. 
   4. We can give -ve with slice: that will extract that part in the given range
4. Nums and Maths
   1. 
   2. More find in the notes
5. Dates: go through code files once
6. Array: []
   1. when we copy array it creates shallow copy means like reference type in heap. deep copy means like copy variables
   2. 4 Methods:
      1. push(6): add element at end
      2. pop(): delete at the end
      3. unshift(9): will add at the start
      4. shift(): delete at the front
   3. Splice and slice
      1. 
   4. Concatation of Array:
7. Object:
   1. We can access object with
      1. Jsuer.mail
      2. Jsuser[“fullname”] -> if we have declared key within “” then we have to compulsory access with point 2
   2. 
      1. TO USE symbol we have to give key like [mySym] and access like a.ii
   3. Concatination of Object:
      1. 
      2. assign method
      3. spread operator
   4. To print keys and values
      1. Object.keys(obj)
      2. Object.values(obj)
      3. Object.entries(obj)
      4. 
   5. To print any thing we generally access with “.”
   6. But we can use this way also
      1. 
8. Function
   1. We can’t use “this” with function. Suppose in the object we have declared uname:”viraj” then we can access like **this.uname**
      1. But doesn’t apply with function
   2. Iife: immediately invoked function expression IIFE
      1. Suppose we want to execute the function immediately like after calling certain function. we want to check the db connection
         1. 
9. Iteration
   1. Forof loop
      1. 
      2. We can’t run forof on object
   2. Map
      1. 
   3. Forin
      1. 
      2. Will not work on Map
   4. forEach
      1. 
      2. 
      3. forEach does not return anything
   5. Filter
      1. A higher-order function is a function that takes one or more functions as arguments (or returns a function as its result). Higher-order functions allow you to abstract away common repetitive tasks, and make your code more concise and reusable.
      2. In JavaScript, there are many built-in higher-order functions, such as map(), filter(), forEach(), and reduce(). These functions can be used to loop through arrays and perform various operations on the elements.
      3. 
      4. Same with filter and forEach we need extra space since it does not return anything
   6. Map
      1. While both map() and forEach() perform similar actions, there are some key differences in their performance. One significant difference is that map() returns a new array, whereas forEach() does not. This means that if you need to create a new array based on the values of an existing array, map() is the better choice
      2. 
   7. Reduce
      1. 