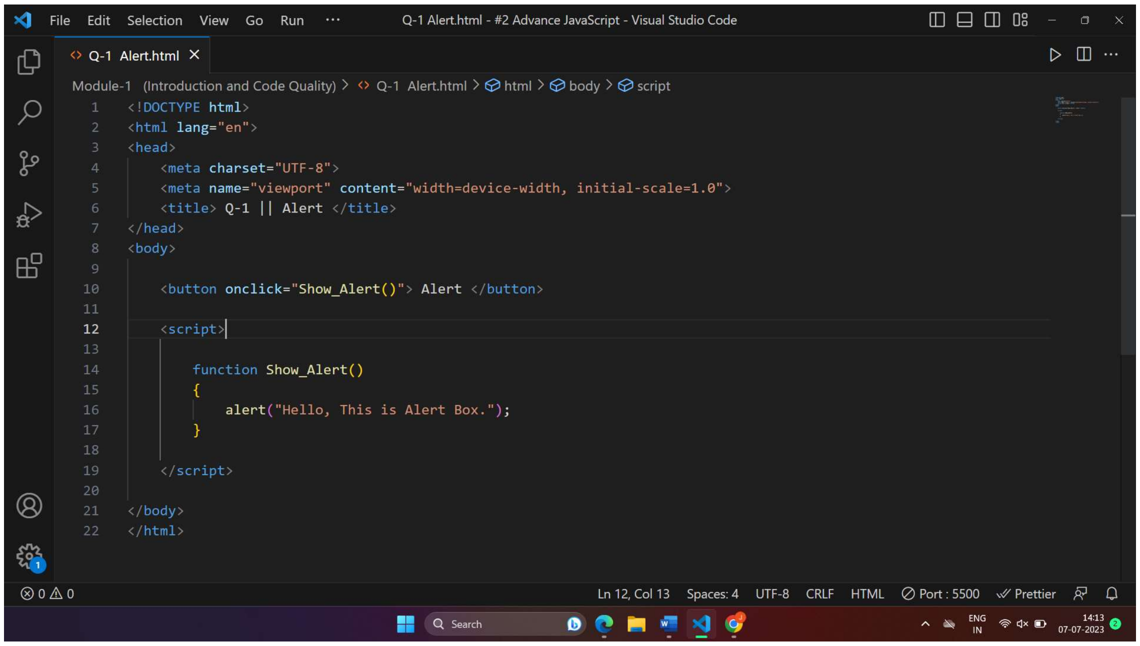
Advance JavaScript

Module : 1 (Introduction and Code Quality)

(1) Write a programme to Show an alert ?



File Edit Selection View Go Run ···
Q-1 Alert.html - #2 Advance JavaScript - Visual Studio Code
Q-1 Alert.html
Module-1 (Introduction and Code Quality) > Q-1 Alert.html > html > body > script
Ln 12, Col 13 Spaces: 4 UTF-8 CRLF HTML Port : 5500 Prettier

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title> Q-1 || Alert </title>
</head>
<body>

    <button onclick="Show_Alert()"> Alert </button>

    <script>

        function Show_Alert()
        {
            alert("Hello, This is Alert Box.");
        }

    </script>

</body>
</html>
```

(2) What will be the result for these expressions ?

1) 5>4

2) “apple” > “pineapple”

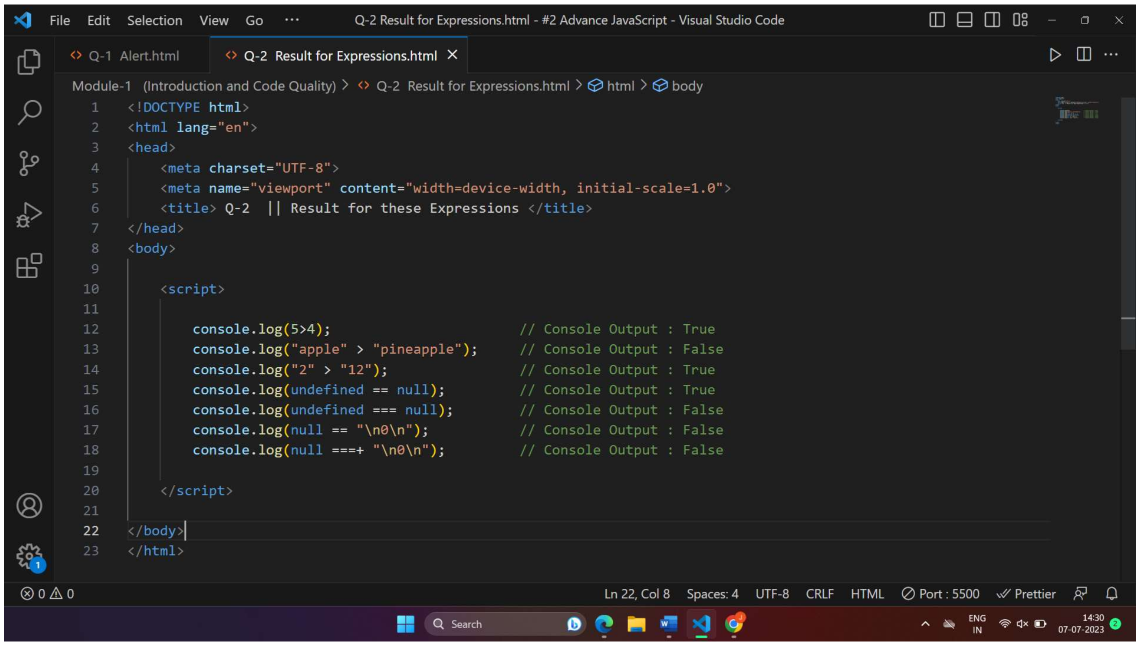
3) “2” > “12”

4) undefined == null

5) undefined === null

6) null == “\n0\n”

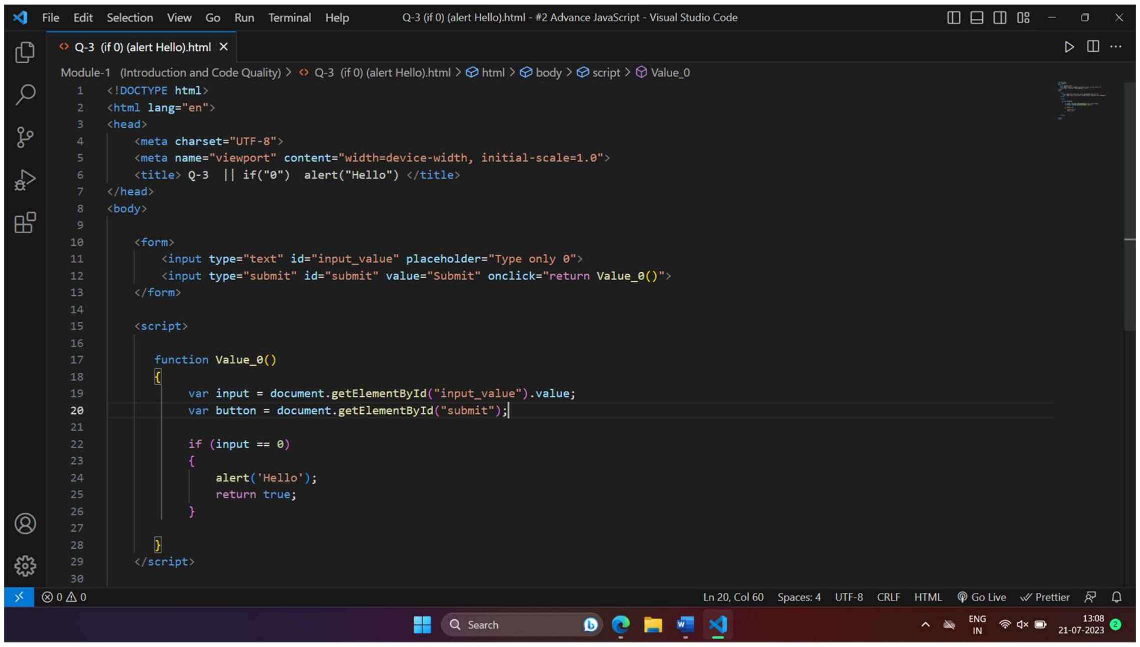
7) null ===+ “\n0\n”



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title> Q-2  || Result for these Expressions </title>
</head>
<body>

    <script>

        console.log(5>4);                       // Console Output : True
        console.log("apple" > "pineapple");     // Console Output : False
        console.log("2" > "12");                // Console Output : True
        console.log(undefined == null);         // Console Output : True
        console.log(undefined === null);        // Console Output : False
        console.log(null == "\n0\n");           // Console Output : False
        console.log(null ===+ "\n0\n");         // Console Output : False

    </script>

</body>
</html>
```

(3) Will alert be shown ?

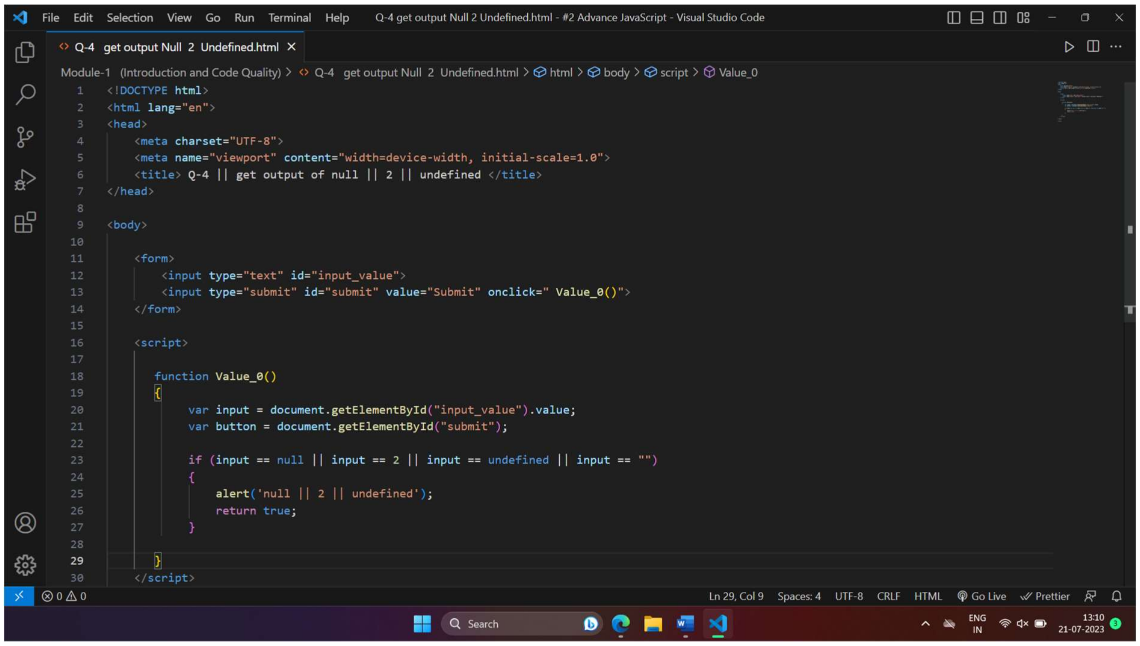
If ("0") { alert("Hello"); }



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title> Q-3  || if("0")  alert("Hello") </title>
</head>
<body>

    <form>
        <input type="text" id="input_value" placeholder="Type only 0">
        <input type="submit" id="submit" value="Submit" onclick="return Value_0()">
    </form>

    <script>

       function Value_0()
       {
            var input = document.getElementById("input_value").value;
            var button = document.getElementById("submit");

            if (input == 0)
            {
                alert('Hello');
                return true;
            }

       }
    </script>

```

(4) What is the code below going to output ?

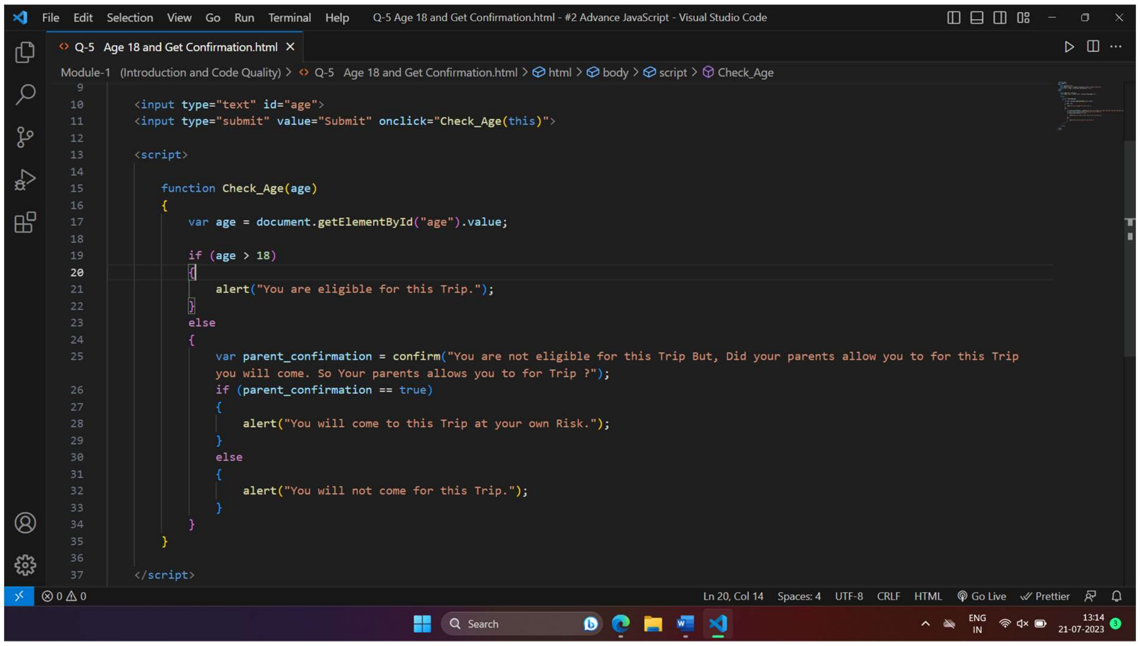
Alert(null || 2 || undefined);


Q-4 get output Null 2 Undefined.html - #2 Advance JavaScript - Visual Studio Code
Module-1 (Introduction and Code Quality) > Q-4 get output Null 2 Undefined.html > html > body > script > Value_0
Ln 29, Col 9 Spaces: 4 UTF-8 CRLF HTML Go Live Prettier

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title> Q-4 || get output of null || 2 || undefined </title>
</head>

<body>

    <form>
        <input type="text" id="input_value">
        <input type="submit" id="submit" value="Submit" onclick=" Value_0()">
    </form>

    <script>

       function Value_0()
       {
            var input = document.getElementById("input_value").value;
            var button = document.getElementById("submit");

            if (input == null || input == 2 || input == undefined || input == "")
            {
                alert('null || 2 || undefined');
                return true;
            }

       }
    </script>
```

(5) The following function returns true if the parameter age is geater than 18. Otherwise it asks for a confirmation and returns its result :

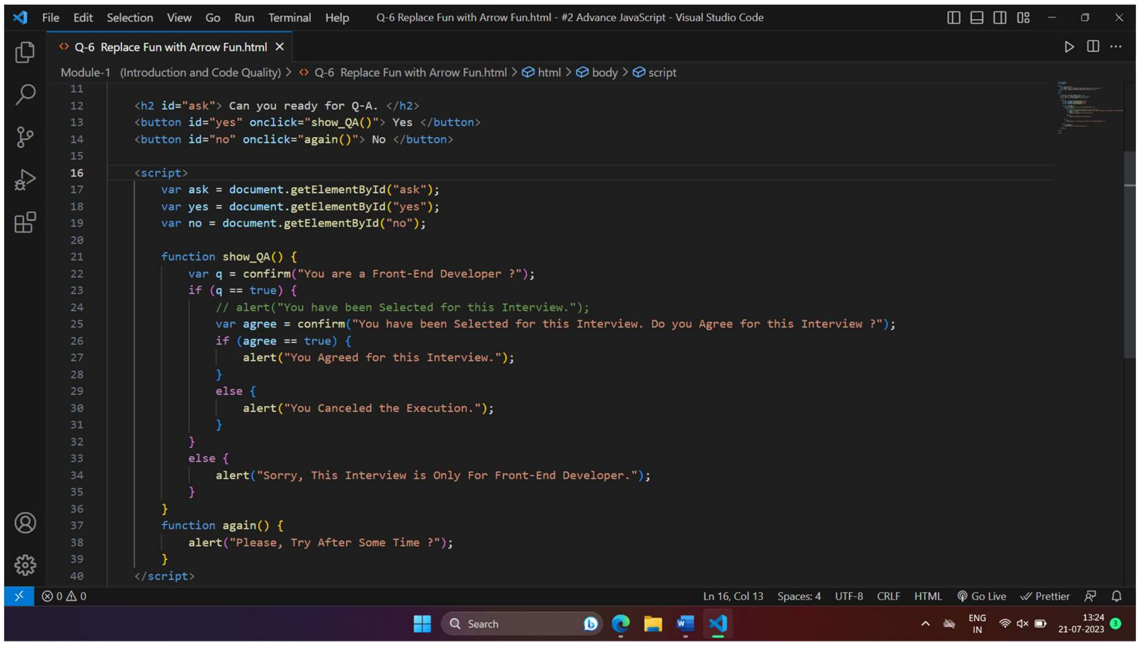
```
function

checkage(age)

{

        If (age>18) { return true; }

else {

        //…return confirm ('did parents allow you?');

        }

}
```


File Edit Selection View Go Run Terminal Help
Q-5 Age 18 and Get Confirmation.html - #2 Advance JavaScript - Visual Studio Code
Q-5 Age 18 and Get Confirmation.html
Module-1 (Introduction and Code Quality) > Q-5 Age 18 and Get Confirmation.html > html > body > script > Check_Age
Ln 20, Col 14 Spaces: 4 UTF-8 CRLF HTML Go Live Prettier
Search
ENG IN
13:14
21-07-2023

```
    <input type="text" id="age">
    <input type="submit" value="Submit" onclick="Check_Age(this)">

    <script>

        function Check_Age(age)
        {
            var age = document.getElementById("age").value;

            if (age > 18)
            {
                alert("You are eligible for this Trip.");
            }
            else
            {
                var parent_confirmation = confirm("You are not eligible for this Trip But, Did your parents allow you to for this Trip
                you will come. So Your parents allows you to for Trip ?");
                if (parent_confirmation == true)
                {
                    alert("You will come to this Trip at your own Risk.");
                }
                else
                {
                    alert("You will not come for this Trip.");
                }
            }
        }

    </script>
```

(6) Replace Function Expressions with arrow fucntions in the code below :

ask (question, yes, no)
{ if (confirm(quetion))yes();
else
no();
}
ask (“Do you agree ?”, function)
{ alert(“You agreed.”); },
function() {
alert(“You canceled the execution.”); }
}

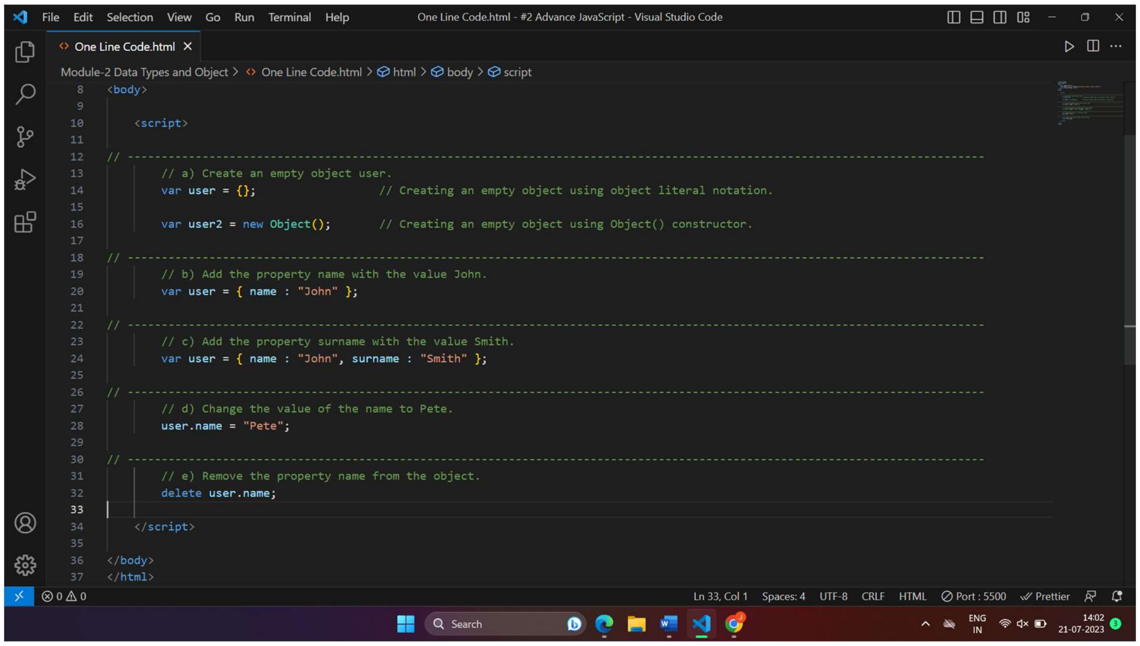

File Edit Selection View Go Run Terminal Help Q-6 Replace Fun with Arrow Fun.html - #2 Advance JavaScript - Visual Studio Code
Q-6 Replace Fun with Arrow Fun.html
Module-1 (Introduction and Code Quality) > Q-6 Replace Fun with Arrow Fun.html > html > body > script
Ln 16, Col 13 Spaces: 4 UTF-8 CRLF HTML Go Live Prettier
Search
ENG IN 13:24 21-07-2023

```

    <h2 id="ask"> Can you ready for Q-A. </h2>
    <button id="yes" onclick="show_QA()"> Yes </button>
    <button id="no" onclick="again()"> No </button>

    <script>
        var ask = document.getElementById("ask");
        var yes = document.getElementById("yes");
        var no = document.getElementById("no");

        function show_QA() {
            var q = confirm("You are a Front-End Developer ?");
            if (q == true) {
                // alert("You have been Selected for this Interview.");
                var agree = confirm("You have been Selected for this Interview. Do you Agree for this Interview ?");
                if (agree == true) {
                    alert("You Agreed for this Interview.");
                }
                else {
                    alert("You Canceled the Execution.");
                }
            }
            else {
                alert("Sorry, This Interview is Only For Front-End Developer.");
            }
        }
        function again() {
            alert("Please, Try After Some Time ?");
        }
    </script>
```

Module : 2 (Data Types and Objects)

- **Write the code, one line for each action :**

a) Create an empty object user.
b) Add the property name with the value John.
c) Add the property surname with the value Smith.
d) Change the value of the name to Pete.
e) Remove the property name from the object.


File Edit Selection View Go Run Terminal Help
One Line Code.html - #2 Advance JavaScript - Visual Studio Code
One Line Code.html
Module-2 Data Types and Object > One Line Code.html > html > body > script
Ln 33, Col 1 Spaces: 4 UTF-8 CRLF HTML Port : 5500 Prettier
Search
ENG IN 14:02 21-07-2023

```html
<body>

    <script>

// -----------------------------------------------------------------------------------------------------------------------------
        // a) Create an empty object user.
        var user = {};                  // Creating an empty object using object literal notation.

        var user2 = new Object();       // Creating an empty object using Object() constructor.

// -----------------------------------------------------------------------------------------------------------------------------
        // b) Add the property name with the value John.
        var user = { name : "John" };

// -----------------------------------------------------------------------------------------------------------------------------
        // c) Add the property surname with the value Smith.
        var user = { name : "John", surname : "Smith" };

// -----------------------------------------------------------------------------------------------------------------------------
        // d) Change the value of the name to Pete.
        user.name = "Pete";

// -----------------------------------------------------------------------------------------------------------------------------
        // e) Remove the property name from the object.
        delete user.name;

    </script>

</body>
</html>
```

- **Is array copied ?**

  let fruits = [“Apples”, “Pear”, “Orange”];

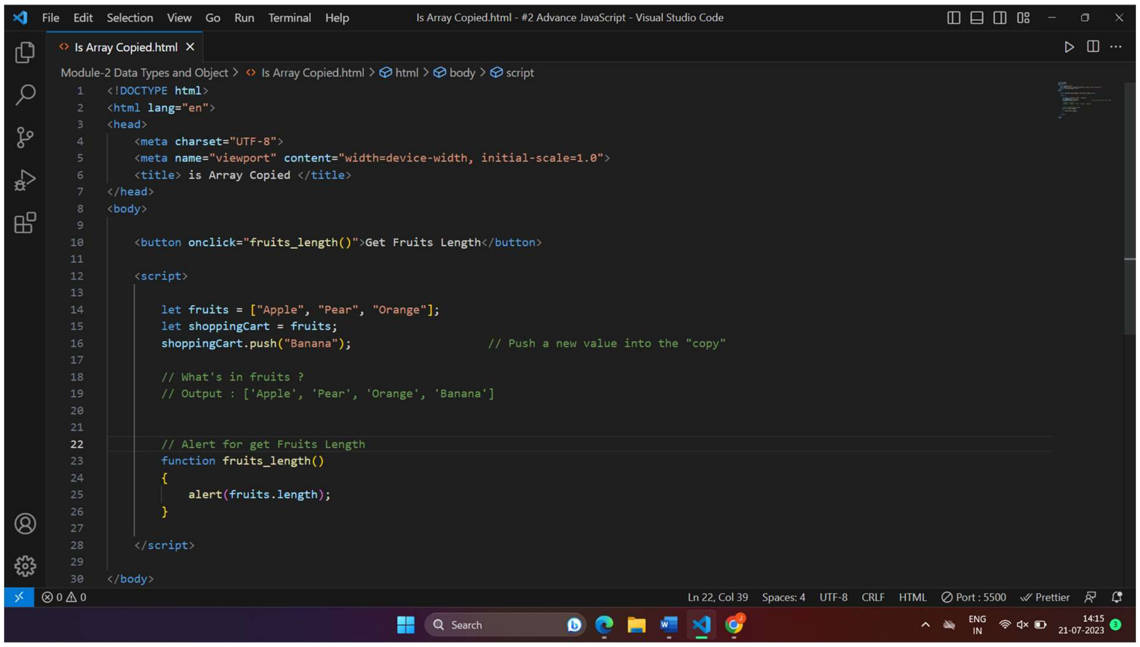
  // push a new value into the “copy”

  let shoppingCart = fruits;

  shoppingCart.push(“Banana”);

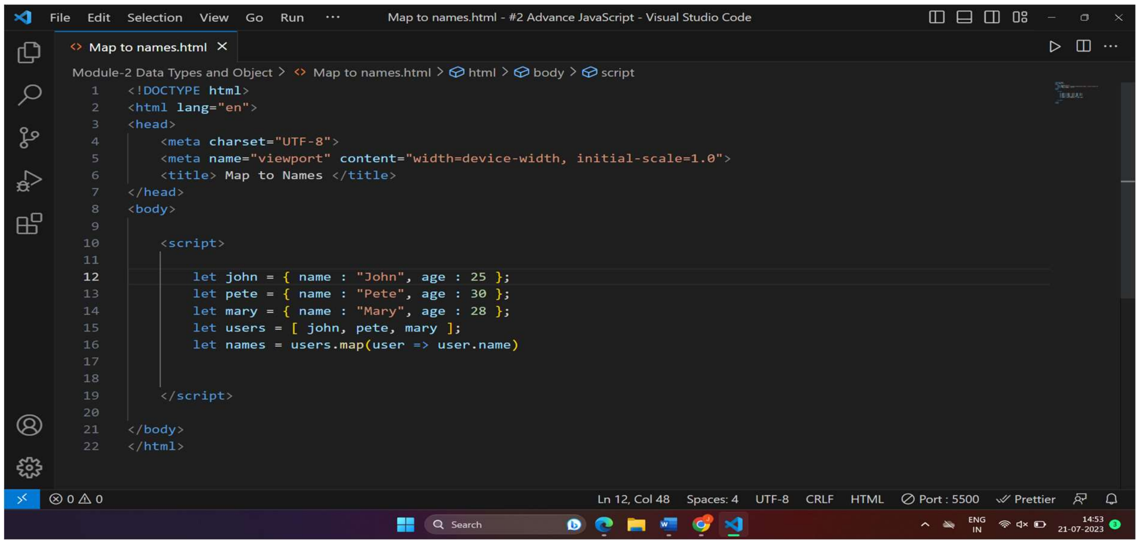
  // what’s in fruits ?

alert(fruits.length);


File Edit Selection View Go Run Terminal Help
Is Array Copied.html - #2 Advance JavaScript - Visual Studio Code
Is Array Copied.html
Module-2 Data Types and Object > Is Array Copied.html > html > body > script
Ln 22, Col 39 Spaces: 4 UTF-8 CRLF HTML Port : 5500 Prettier

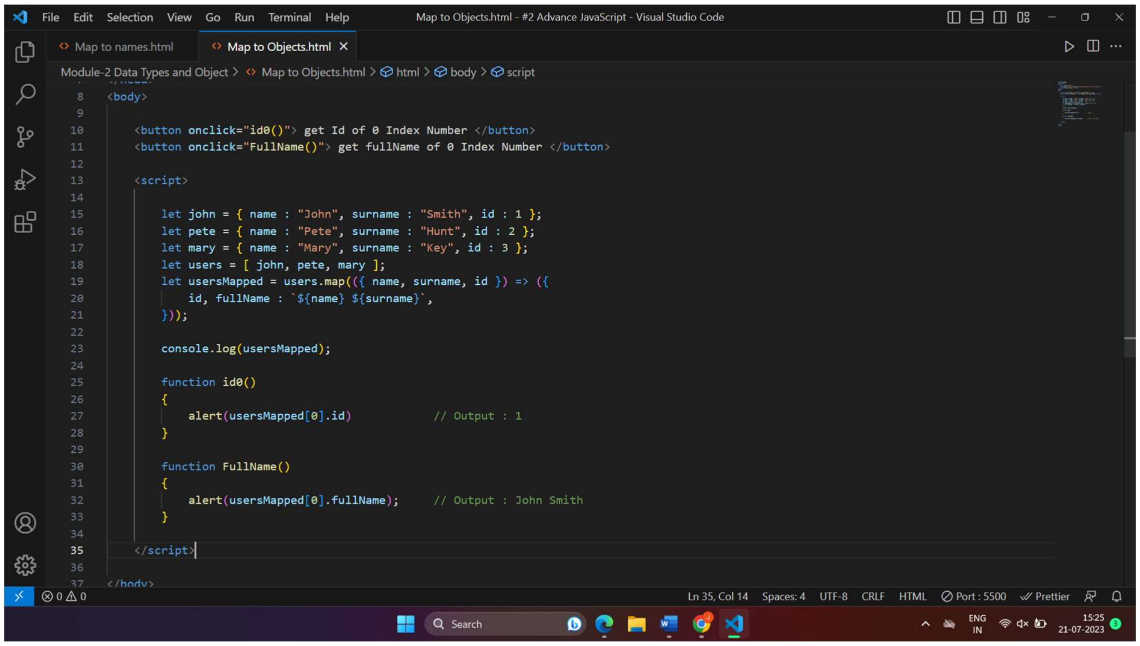
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title> is Array Copied </title>
</head>
<body>

    <button onclick="fruits_length()">Get Fruits Length</button>

    <script>

        let fruits = ["Apple", "Pear", "Orange"];
        let shoppingCart = fruits;
        shoppingCart.push("Banana");                    // Push a new value into the "copy"

        // What's in fruits ?
        // Output : ['Apple', 'Pear', 'Orange', 'Banana']


        // Alert for get Fruits Length
        function fruits_length()
        {
            alert(fruits.length);
        }

    </script>

</body>
```

- **Map to names**
  let john = { name: “John”, age: 25 };
  let pete = { name:”Pete”, age: 30 };
  let mary = { name: “Mary”, age: 28 };
  let users = [john, pete, mary];
  let names = /* … your code */
  alert(names);   // John, Pete, Mary


File Edit Selection View Go Run ...
Map to names.html - #2 Advance JavaScript - Visual Studio Code
Map to names.html
Module-2 Data Types and Object > Map to names.html > html > body > script
Ln 12, Col 48 Spaces: 4 UTF-8 CRLF HTML Port : 5500 Prettier

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title> Map to Names </title>
</head>
<body>

    <script>

        let john = { name : "John", age : 25 };
        let pete = { name : "Pete", age : 30 };
        let mary = { name : "Mary", age : 28 };
        let users = [ john, pete, mary ];
        let names = users.map(user => user.name)


    </script>

</body>
</html>
```

- **Map to objects**
  Let john = { name: “John”, surname: “Smith”, id: 1 };
  Let pete = { name: “Pete”, surname: “Hunt”, id: 2 };
  Let mary = { name: “Mary”, surname: “Key”, id: 3 };
  Let users = [john, pete, mary];
  let users Mapped = /* …your code …*/
  /*
  usersMapped = [
  { fullName: “John Smith”, id; 1 },
  { fullName: “Pete Hunt”, id; 2 },
  { fullName: “Mary Key”, id; 3 }
  ]
  ***/**
  alert(usersMapped[0].id)   // 1
  alert(usersMapped[0].fullName)   // John Smith


File Edit Selection View Go Run Terminal Help
Map to Objects.html - #2 Advance JavaScript - Visual Studio Code
Map to names.html
Map to Objects.html
Module-2 Data Types and Object > Map to Objects.html > html > body > script
Ln 35, Col 14 Spaces: 4 UTF-8 CRLF HTML Port : 5500 Prettier
Search
ENG IN 15:25 21-07-2023

```
<body>

    <button onclick="id0()"> get Id of 0 Index Number </button>
    <button onclick="FullName()"> get fullName of 0 Index Number </button>

    <script>

        let john = { name : "John", surname : "Smith", id : 1 };
        let pete = { name : "Pete", surname : "Hunt", id : 2 };
        let mary = { name : "Mary", surname : "Key", id : 3 };
        let users = [ john, pete, mary ];
        let usersMapped = users.map(({ name, surname, id }) => ({
            id, fullName : `${name} ${surname}`,
        }));

        console.log(usersMapped);

        function id0()
        {
            alert(usersMapped[0].id)            // Output : 1
        }

        function FullName()
        {
            alert(usersMapped[0].fullName);     // Output : John Smith
        }

    </script>

</body>
```

- **Sum the properties There is a salaries object with arbitrary number of salaries. Write the function sumSalaries(salaries)**

that returns the sum of all salaries using Object. values and the for..of loop. If salaries is empty, then the result must be ().

let salaries = {

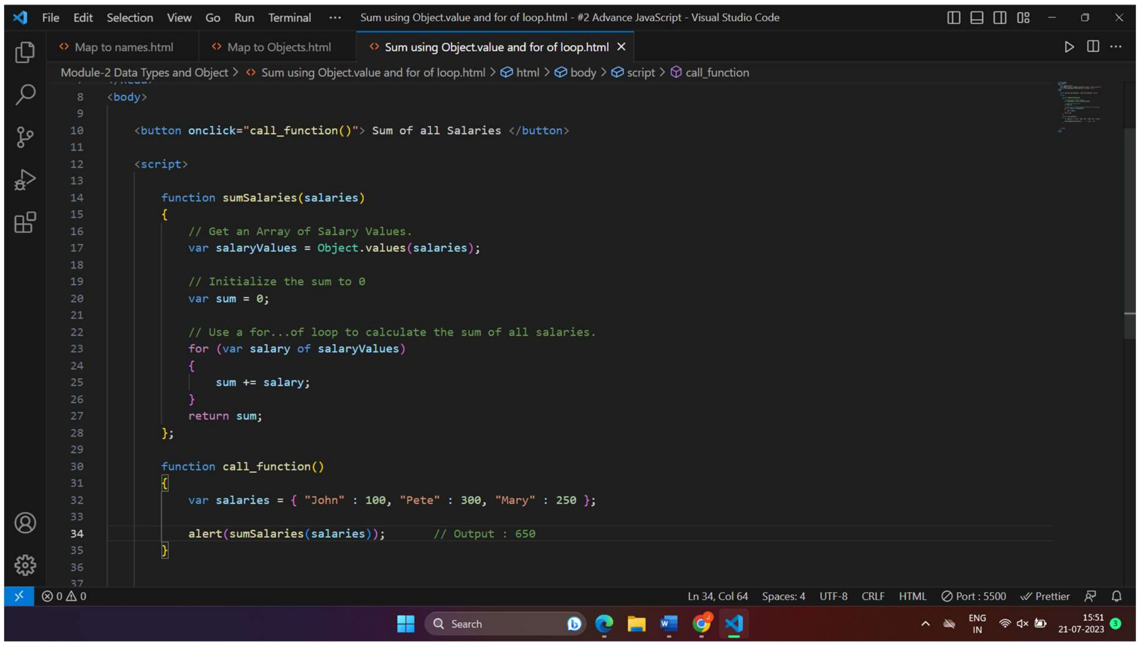
“John”: 100,

“Pete”: 300,

“Mary”: 250

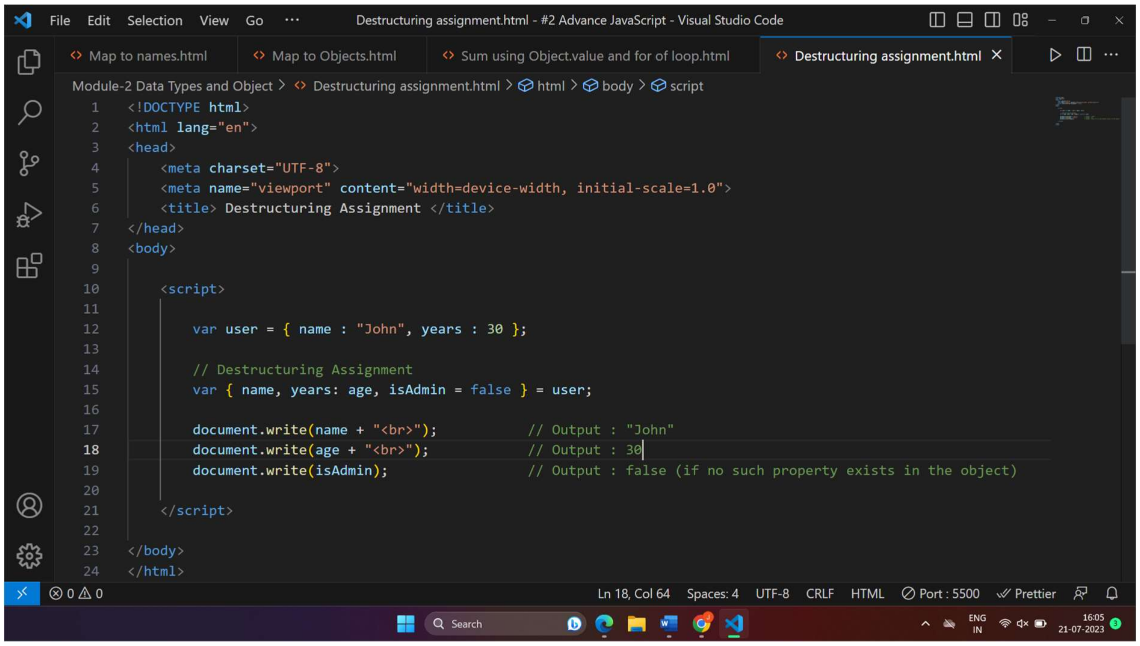
};

alert(sumSalaries(salaries)); // 650


File Edit Selection View Go Run Terminal ···
Sum using Object.value and for of loop.html - #2 Advance JavaScript - Visual Studio Code
Map to names.html
Map to Objects.html
Sum using Object.value and for of loop.html
Module-2 Data Types and Object > Sum using Object.value and for of loop.html > html > body > script > call_function
Ln 34, Col 64 Spaces: 4 UTF-8 CRLF HTML Port : 5500 Prettier
Search
ENG IN 15:51 21-07-2023

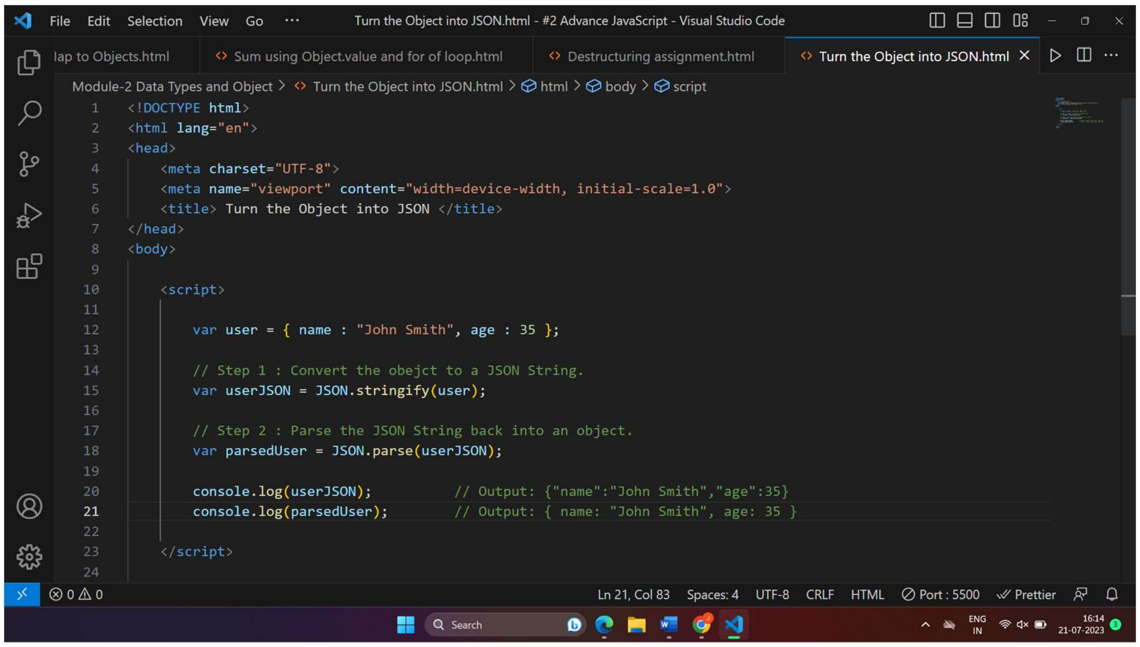
```
<body>

    <button onclick="call_function()"> Sum of all Salaries </button>

    <script>

        function sumSalaries(salaries)
        {
            // Get an Array of Salary Values.
            var salaryValues = Object.values(salaries);

            // Initialize the sum to 0
            var sum = 0;

            // Use a for...of loop to calculate the sum of all salaries.
            for (var salary of salaryValues)
            {
                sum += salary;
            }
            return sum;
        };

        function call_function()
        {
            var salaries = { "John" : 100, "Pete" : 300, "Mary" : 250 };

            alert(sumSalaries(salaries));        // Output : 650
        }
```

- **Destructing assignment We have an object : Write the Destructing assignment that reads :**
  a) Name property into the variable name.
  b) Year’s property into the variable age.
  c) isAdmin property into the variable isAdmin (false, if no such property)
  d) let user = { name: “John”, age: 30 };



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title> Destructuring Assignment </title>
</head>
<body>

    <script>

        var user = { name : "John", years : 30 };

        // Destructuring Assignment
        var { name, years: age, isAdmin = false } = user;

        document.write(name + "<br>");          // Output : "John"
        document.write(age + "<br>");           // Output : 30
        document.write(isAdmin);                // Output : false (if no such property exists in the object)

    </script>

</body>
</html>
```

- **Turn the object into JSON and back Turn the user into JSON and then read it back into another variable.**
  User = { name: “John Smith”, age: 35 };



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title> Turn the Object into JSON </title>
</head>
<body>

    <script>

        var user = { name : "John Smith", age : 35 };

        // Step 1 : Convert the obejct to a JSON String.
        var userJSON = JSON.stringify(user);

        // Step 2 : Parse the JSON String back into an object.
        var parsedUser = JSON.parse(userJSON);

        console.log(userJSON);          // Output: {"name":"John Smith","age":35}
        console.log(parsedUser);        // Output: { name: "John Smith", age: 35 }

    </script>

```

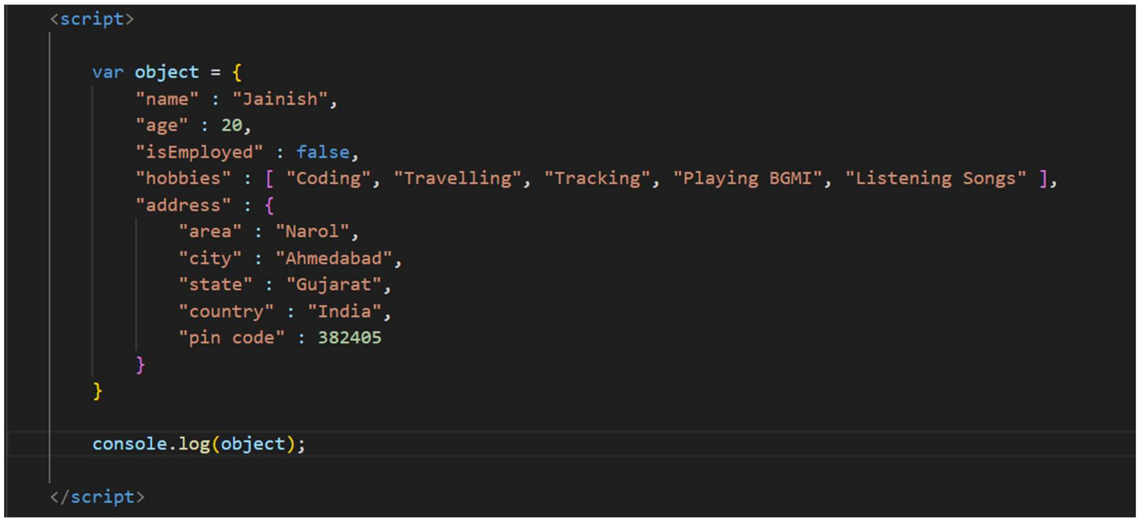
Module : 4 (New Request)

- **What is JSON**

➔ JSON stands for “JavaScript Object Notation”, and it is a lightweight data interchange format. JSON is commonly used for transmitting data between a server and a web application as an alternative to XML.

In simple terms, JSON provides a standardized way to represent data in a format that is both human-readable and machine-readable. It is based on key-value pairs, where each key is a string and each value can be a String, Number, Boolean, Array or Another JSON object.

Example :



```
<script>

    var object = {
        "name" : "Jainish",
        "age" : 20,
        "isEmployed" : false,
        "hobbies" : [ "Coding", "Travelling", "Tracking", "Playing BGMI", "Listening Songs" ],
        "address" : {
            "area" : "Narol",
            "city" : "Ahmedabad",
            "state" : "Gujarat",
            "country" : "India",
            "pin code" : 382405
        }
    }

    console.log(object);

</script>
```

In this above JSON example, we have a simple object with various data types :

- ❖ “name” : “Jainish”   :- Key “name” with a String value “Jainish”
- ❖ “age” : 30   :- Key “age” with a Numeric value 30
- ❖ “isEmployed” : false   :- Key “isEmployed” with a Boolean value False.
- ❖ “hobbies” : [“Coding”, “Travelling”, “Tracking”, “Playing BGMI”, “Listening Songs”]   :- Key “hobbies" with an Array of Strings as its Value.
- ❖ “address” : { “area” : “Narol”, “city” : “Ahmedabad”, “state” : “Gujarat”, “country” : “India”, “pin code” : 382405 }   :- Key “address” with an Object as its value containing keys “area”, “city”, “state”, “country” and “pin code”.

JSON is widely used in Web-Development for APIs, Configuration files and Data interchange due to its simplicity, readability and compatibility with different programming languages. It has become a standard format for data exchange and communication on the web.

- **What is promises**

➔ JavaScript Promise Object : A JavaScript Promise object contains both the producing code and calls to the consuming code.

Promise Object Properties

A JavaScript Promise Object can be :

(1) Pending : The intial state when the task is still in progress.
(2) Fulfilled (Resolved) : The task was successful and the promise has a result (data).
(3) Rejected : The task encountered an error and the promise has an error message.

A Simple analogy for promises could be ordering food from a restaurant. You place an order (creating a promise) and the restaurant can do three things :

(1) Preparing your order (Pending) : The restaurant is working on your order.
(2) Delivering your order (Fulfilled) : The order is successfully delivered to you.
(3) Apologizing for a mistake (Rejected) : The restaurant informs you that they cannot fulfil your order due to some issue.

Module : 4 (JavaScript Essentials)

- **What is JavaScript Output method ?**

➔ In JavaScript, There are several methods available to display output, which allow you to show information or results to the user.
JavaScript can "display" data in different ways :
(1) Writing into an HTML element, using "innetHTML".
(2) Writing into the HTML output using "document.write()".
(3) Writing into the browser console, using "console.log()".
(4) Writing into an alert box, using "alert()".
(5) Writing into an confirm box, using "confirm()".
(6) Writing into an prompt box, using "prompt ()".

- **How to used JavaScript Output method ?**

➔ Some JavaScript Output Methods :

(1) console.log()

Use this method to display output in the browser console. It' perfect for debugging and checking the values of variables or messages during development.

(2) document.write()

Use this method to write content directly to the web page. It's best for simple demonstrations or testing purposes, but avoid using it after the page has loaded, as it can overwrite the entire document.

(3) alert()

Use this method to show a pop-up message to the user with an "OK" button. It's useful for providing important notifications or messages.

(4) innerHTML()

Use this property to update the HTML content of an element on the web page dynamically. It's commonly used for displaying information based on user interactions or data from JavaScript.

- **How to used JavaScript Events to do all examples ?**

➔ JavaScript events are used to respond to user interactions or actions that occur on a web page. You can attach event handlers to specific HTML elements or document-wide events to execute JavaScript code when the events occur.

(1) console.log() using Event :
We can use an event listener on a button to log a message to the console when the button is clicked.

(2) **alert() using Event :**
We can show an alert when a link is clicked.

(3) document.write() using Event :
We can use an event listener on a button to write content to the document when the button is clicked.

(4) innerHTML using Event :
We can update content when a button clicked.

In each example, we select the HTML element with JavaScript (‘getElementById’, etc.) and use ‘addEvenListener’ to attach an event handler function to the element. When the specified event (Example : “click”) occurs, the associated function is executed, provided the desired functionality.

Events are fundamentals for creating interactive and responsive web pages. You can use various types of events, such as click, mouseover, mouseout, keydown, keyup, keypress, submit, blur, etc., to handle user interactions and execute JavaScript code accordingly.