Module 6) JAVASCRIPT BASIC & DOM (Basic logic Question)

Q.1 What is JavaScript? How to use it?

- Javascript is a scripting language. It was designed to add interactivity to HTML Pages.
- Javascript is an interpreted language that means the script executes without preliminary compilation.
- It's code directly embedded to the HTML Page. It works for all major browsers.
- Javascript is a client side, interpreted, Object Oriented, high level scripting language which cannot run stand alone.
- It was created By Netscape in 1995, evolved from Netscape's Live Script.
- Javascript works with HTML and CSS (Cascading Style Sheets).
- You can attach javascript file to html file using two methods:
  Internal and external file.

- **Internal javascript**
- **It can be written in head tag or in body tag both ways.**

```
<html>
<head>
<title></title>
<script src=”path”>
– write your code here –
 </script>
</head>
<body></body>
</html>
```

```
<html>
<head>
<title></title>
</head>
<body>
<script src=”path”>
– write your code here –
 </script>
</body>
</html>
```

- **External file**
- **Make an external js file and attach it in html.**

```
<html>
<head>
<title></title>
<script src=”script.js”></script>
```

```
</head>
<body></body>
</html>
```

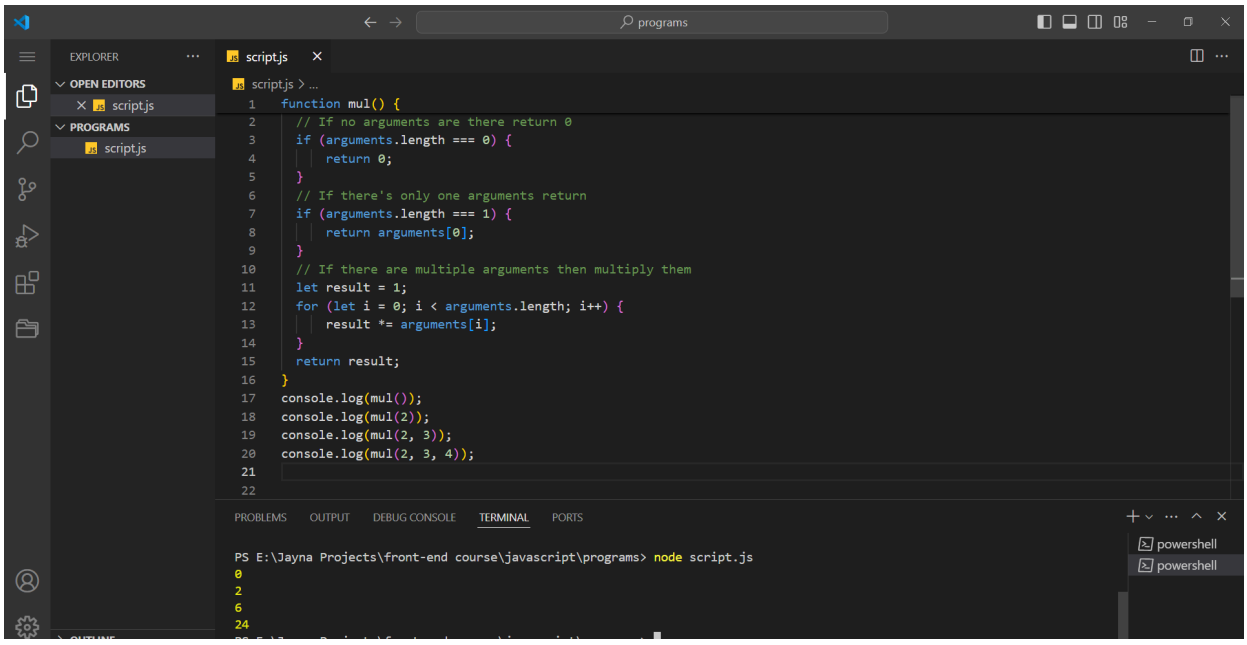
Q.2 How many types of Variables in JavaScript?

- Variables are containers for storing data.
- Javascript variables can be define in 4 ways:
  Using **const** = use const if the value should not be changed.
  Using **var** = Only use var if you MUST support old browsers.
  Using **let** = Only use let if you can't use const.
- The var keyword was used in all JavaScript code from 1995 to 2015.
- The let and const keywords were added to JavaScript in 2015.
- The var keyword should only be used in code written for older browsers.

Q.3 Define a Data Types in js?

- In programming, data types are an important concept.
- To be able to operate on variables, it is important to know something about the type.
- JavaScript has 8 Data Types:
- **String,Number,Bigint,Boolean,Undefined,Null,Symbol,Object**.

Q.4 Write a mul Function Which will Work Properly When invoked With Following Syntax.

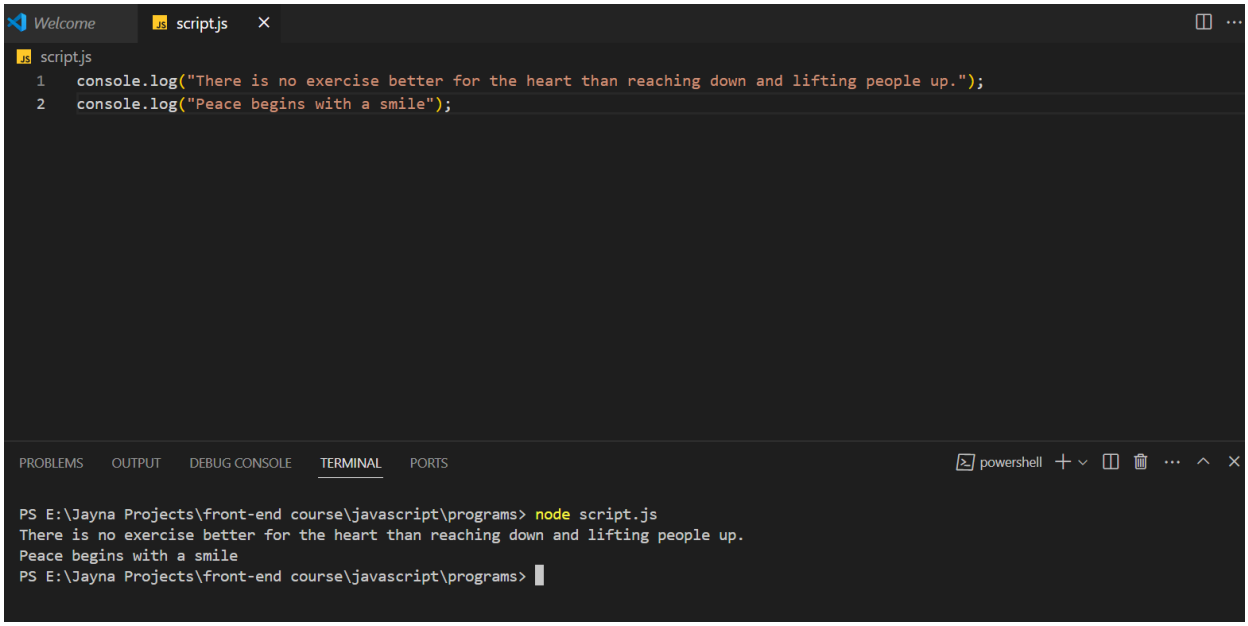


Q.5 What is the difference between undefined and undeclared in JavaScript?

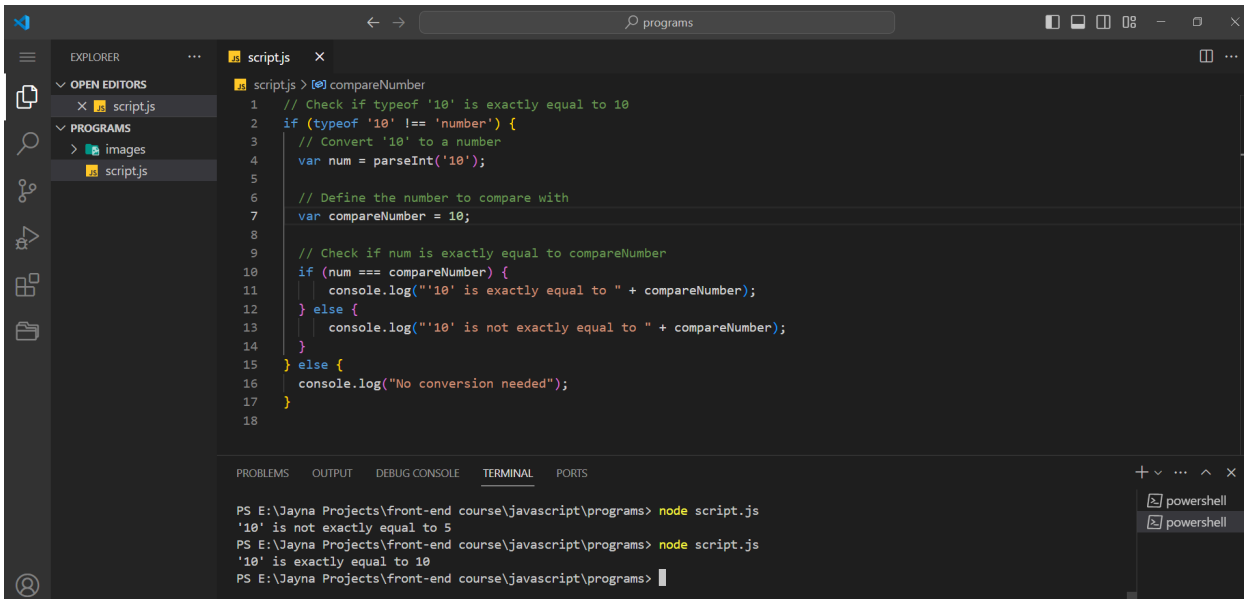
- **Undefined**: It occurs when a variable has been declared but has not been assigned any value. Undefined is not a keyword.

- **Undeclared**: It occurs when we try to access any variable that is not initialized or declared earlier using the var or const keyword.

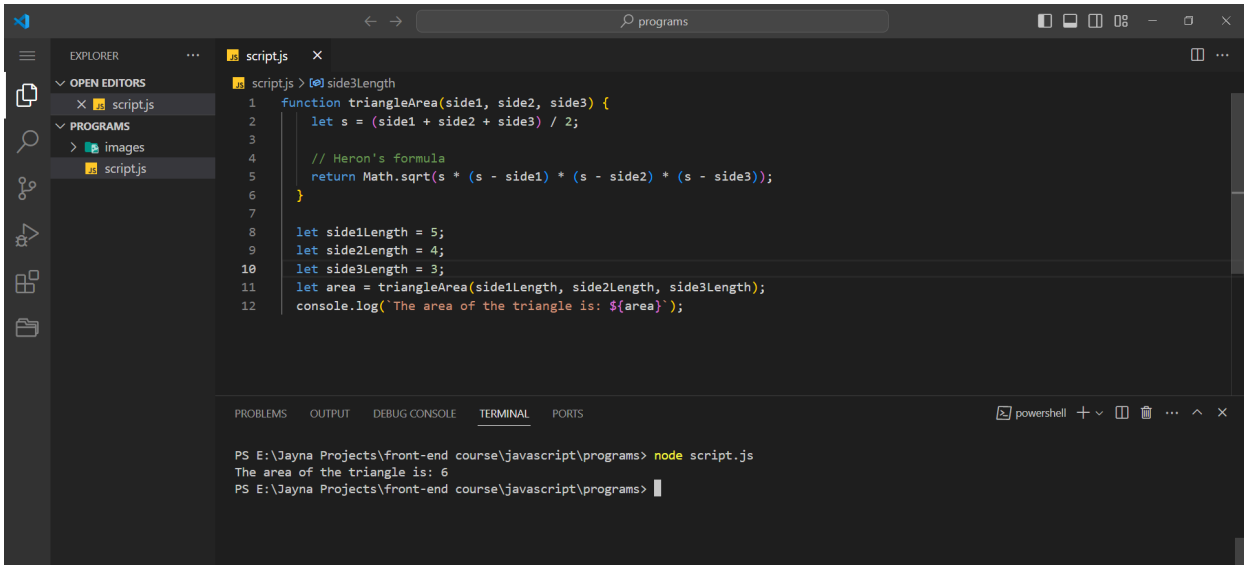
Q.6 Using console.log() print out the following statement: The quote 'There is no exercise better for the heart than reaching down and lifting people up.' by John Holmes teaches us to help one another. Using console.log() print out the following quote by Mother Teresa:



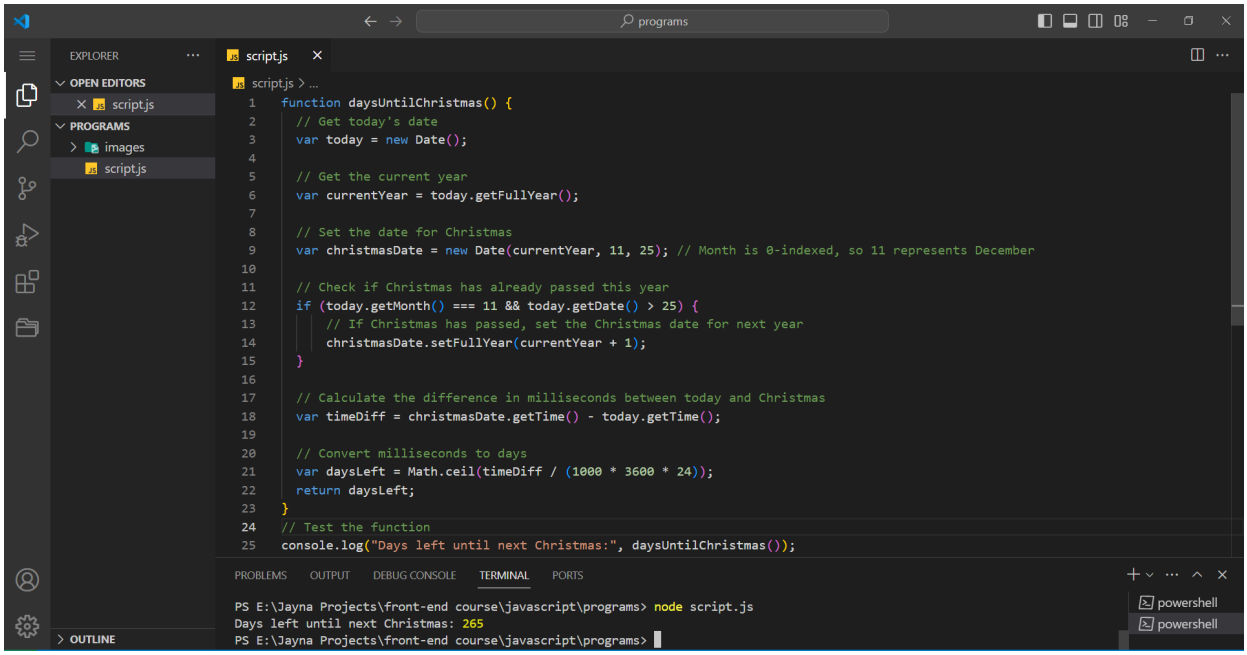
Q.7 Check if typeof '10' is exactly equal to 10. If not make it exactly equal?



Q.8 Write a JavaScript Program to find the area of a triangle?



Q.9 Write a JavaScript program to calculate days left until next Christmas?



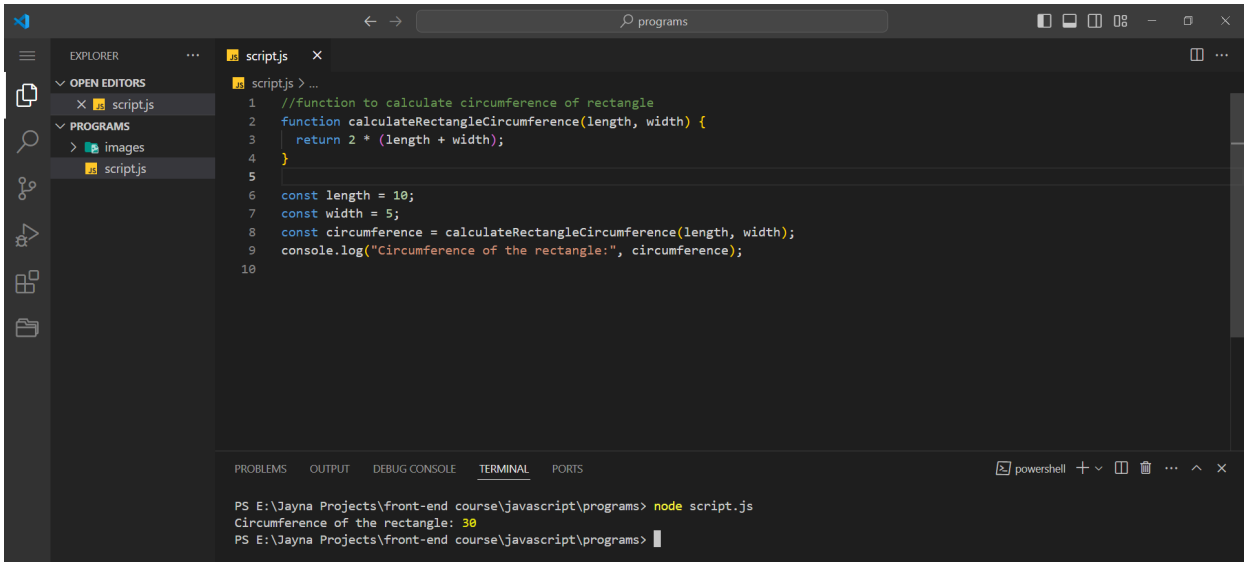
Q.10 What is a Condition Statement?

- Conditional statement is very often used when you write code and you want to perform different actions for different decisions.
- In JavaScript these are the following conditional statements:

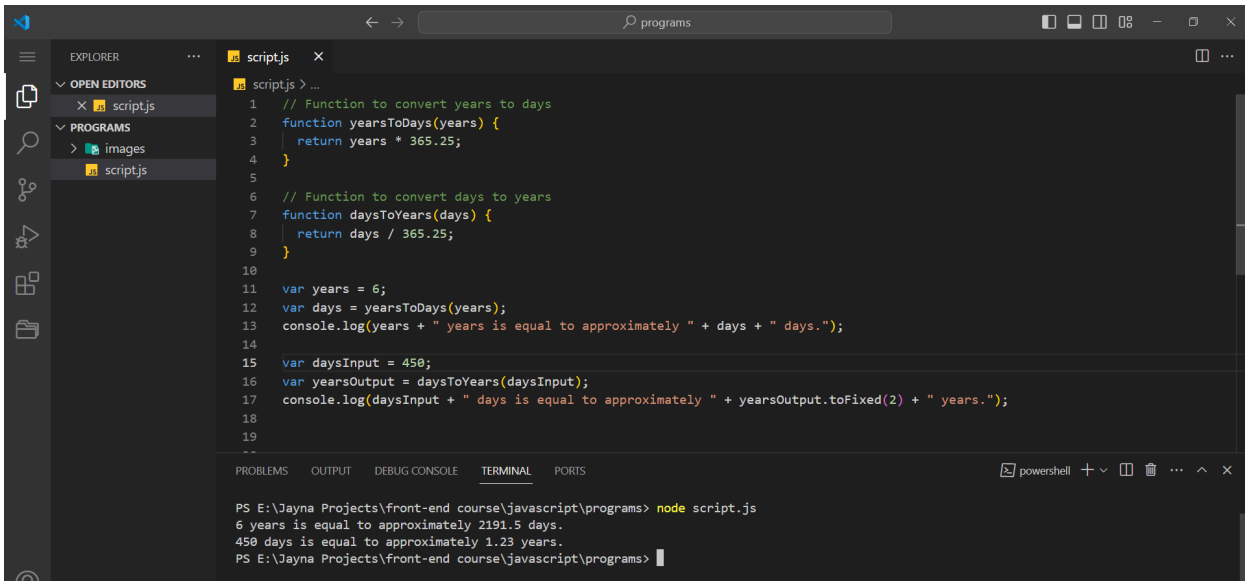
- Use **if** to specify a block of code to be executed, if a specified condition is true

- Use **else** to specify a block of code to be executed, if the same condition is false
- Use **else if** to specify a new condition to test, if the first condition is false
- Use **switch** to specify many alternative blocks of code to be executed.

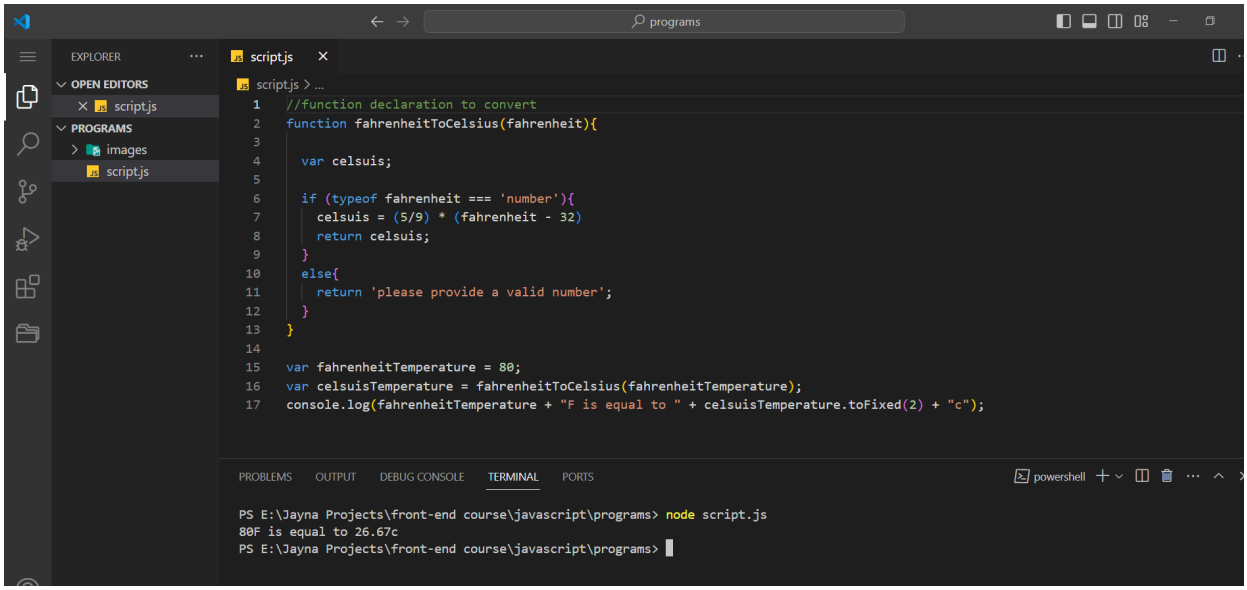
Q.11 Find circumference of Rectangle formula : C = 4 * a ?



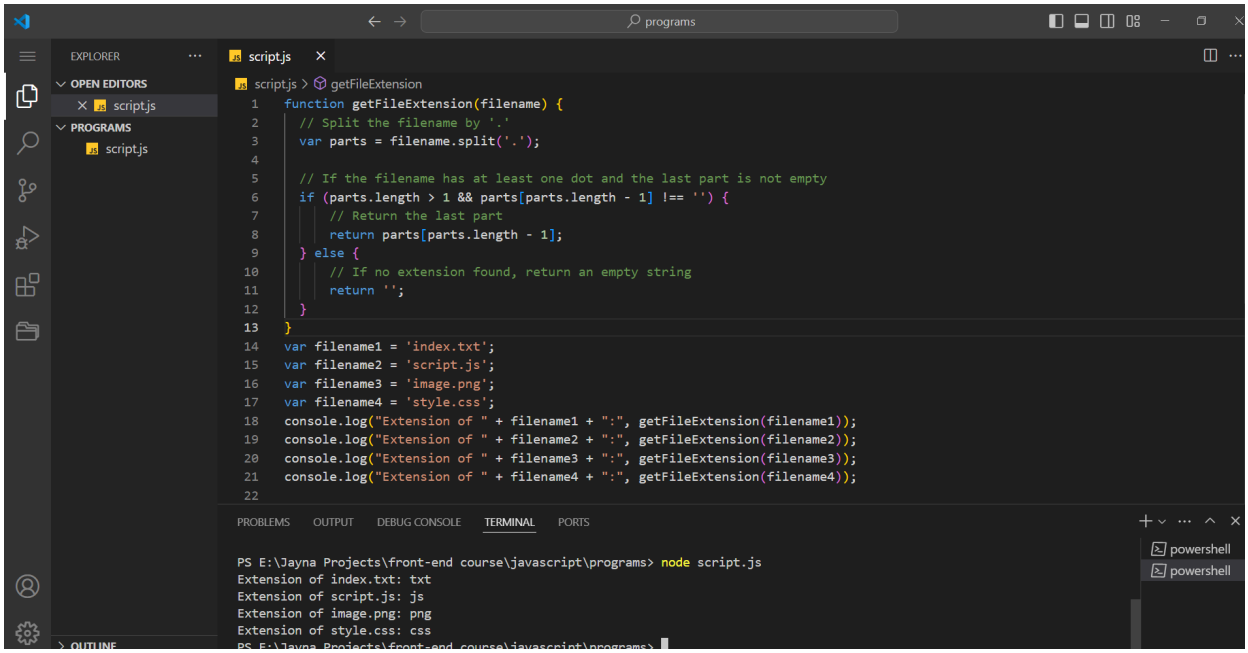
Q.12 WAP to convert years into days and days into years?



Q.13 Convert temperature Fahrenheit to Celsius? (Conditional logic Question)



Q.14 Write a JavaScript exercise to get the extension of a filename.?



Q.15 What is the result of the expression (5 > 3 && 2 < 4)?

- The expression (5>3 && 2<4) is **true**. Because the condition 5>3 is true and 2<4 is true.
- So, the AND operator returns true because both the conditions are true.

Q.16 What is the result of the expression (true && 1 && "hello")?

- The result of the expression is **“hello**”.

- Javascript evaluates && expression from left to right. If all operands are truthy, it returns the value of the last operand.
- So in this case:
    - true is truthy.
    - 1 is truthy.
    - “hello” is also truthy.

Q.17 What is the result of the expression true && false || false && true?

- The result of the expression is **false**.
- Javascript evaluates logical AND (&&) expression before logical OR (||) expression.
- Evaluation step by step:
    - True && false: This evaluates false because one of the operands is false.
    - False && true: This evaluates false for the same reason.
    - False || false: This evaluates false because both operands are false.
- So, the final result is false.

Q.18 What is a Loop and Switch Case in JavaScript define that ?

- In JavaScript, a loop is a programming construct that allows you to execute a block of code repeatedly until a certain condition is met.
- There are several types of loops in JavaScript, including the for loop, while loop, and do-while loop.

- **for loop**: The for loop is used when you know the number of times you want to execute a block of code. It consists of three parts: initialization, condition, and iteration. The loop continues to execute as long as the condition evaluates to true.

  Syntax:
  ```
  for (initialization; condition; iteration) {
   // Code to be executed
  }
  ```

- **while loop:** The while loop is used when you want to execute a block of code as long as a condition is true. The loop evaluates the condition before executing the block of code.

  Syntax:
  ```
  while (condition) {
  // Code to be executed
  }
  ```

- **do-while loop:** The do-while loop is similar to the while loop, but it always executes the block of code at least once before checking the condition. The loop continues to execute as long as the condition evaluates to true.

  Syntax:
  ```
  do {
  // Code to be executed
  } while (condition);
  ```

- **Switch case** is another control flow statement in JavaScript used to perform different actions based on different conditions. It provides a cleaner and more organized alternative to using multiple if...else statements.

  Syntax:
  switch (expression) {
  case value1:
  value1
  break;
  case value2:
  value2
  break;
  default:
  }

Q.19 What is the use of is Nan function?

- In javascript NAN stands for “Not a number”.
- It is a special value that represents the result of an operation that cannot produce a meaningful numeric result.
- The NAN value is returned when:
  - Performing arithmetic operations involving non-numeric values or invalid calculations.
  - Converting non-numeric strings to numbers using parseInt() or parseFloat() functions when the string cannot be parsed as a valid number.

Q.20 What is the difference between && and || in JavaScript?

- In javascript && and || are both logical operators used for boolean operations, but they behave differently:

  &&(Logical AND):
  - The && operator returns true if both operands are true, otherwise it returns false.
  - It short-circuits: if the left operand is false, the right operand is not evaluated because the result will always be false.
  - Example: true && true returns true, true && false returns false, false && true returns false, false && false returns false.

  || (Logical OR):

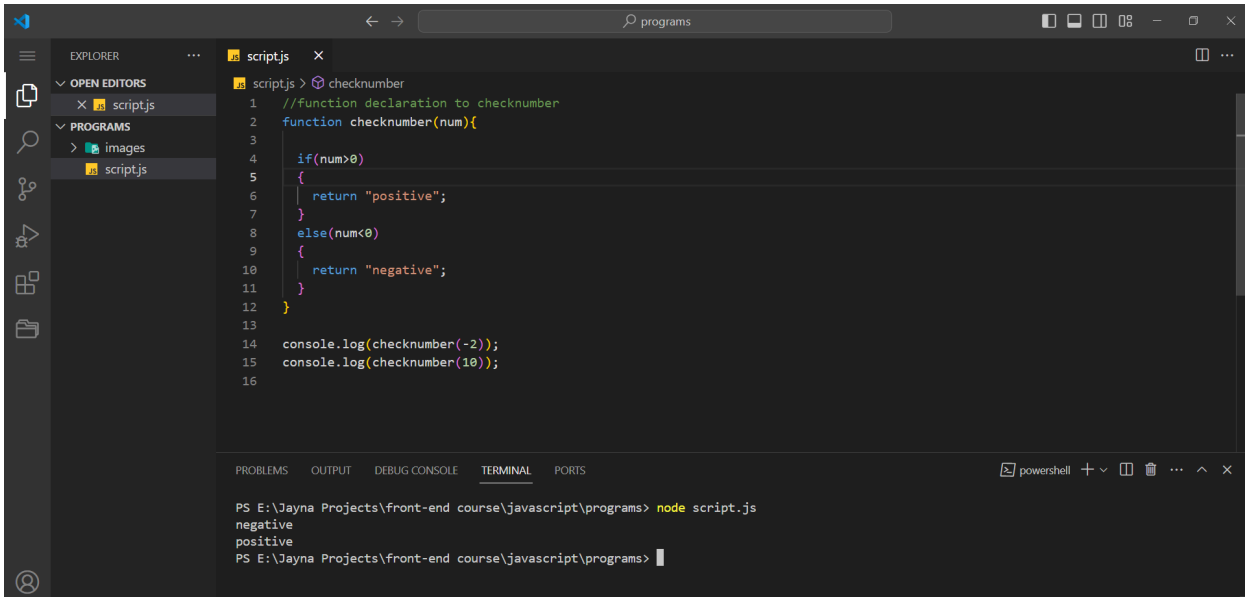
  - The || operator returns true if at least one of the operands is true, otherwise it returns false.
  - It short-circuits: if the left operand is true, the right operand is not evaluated because the result will always be true.
  - Example: true || true returns true, true || false returns true, false || true returns true, false || false returns false.

Q.21 What is the use of Void (0)?

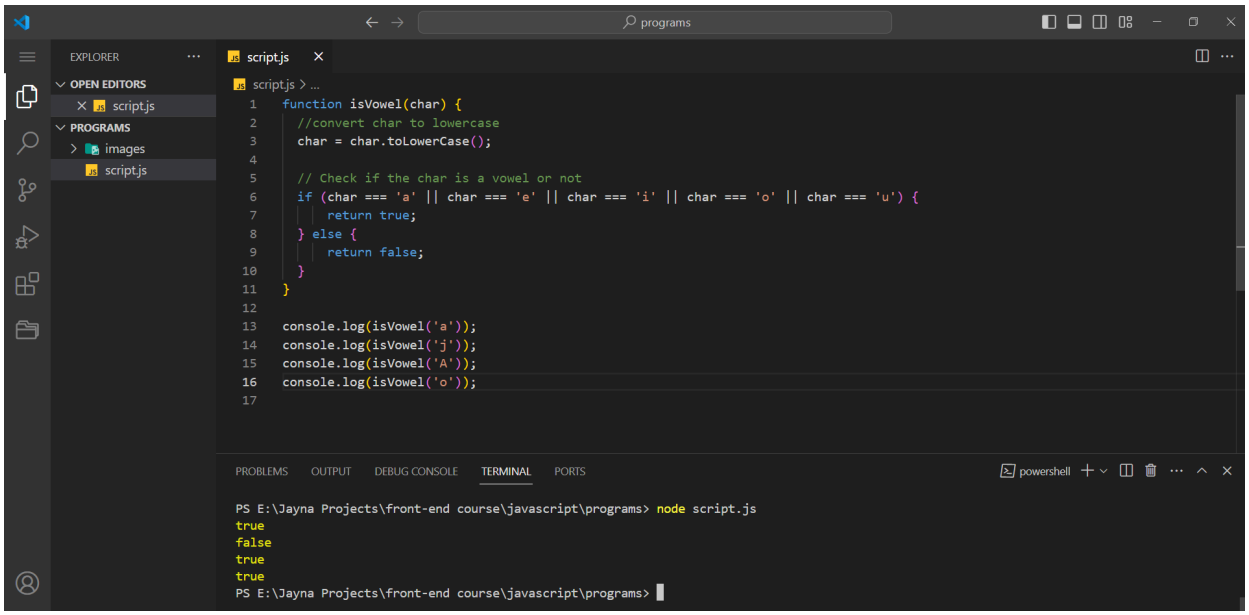
- In javascript void(0) is an expression that evaluates to undefined.
- It's often used to return undefined from an expression context.

- The primary use case of void(0) is to create an expression that returns undefined without explicitly using the undefined keyword.
- It can be useful in scenarios where you want to ensure that a function or event handler doesn't inadvertently return a value.

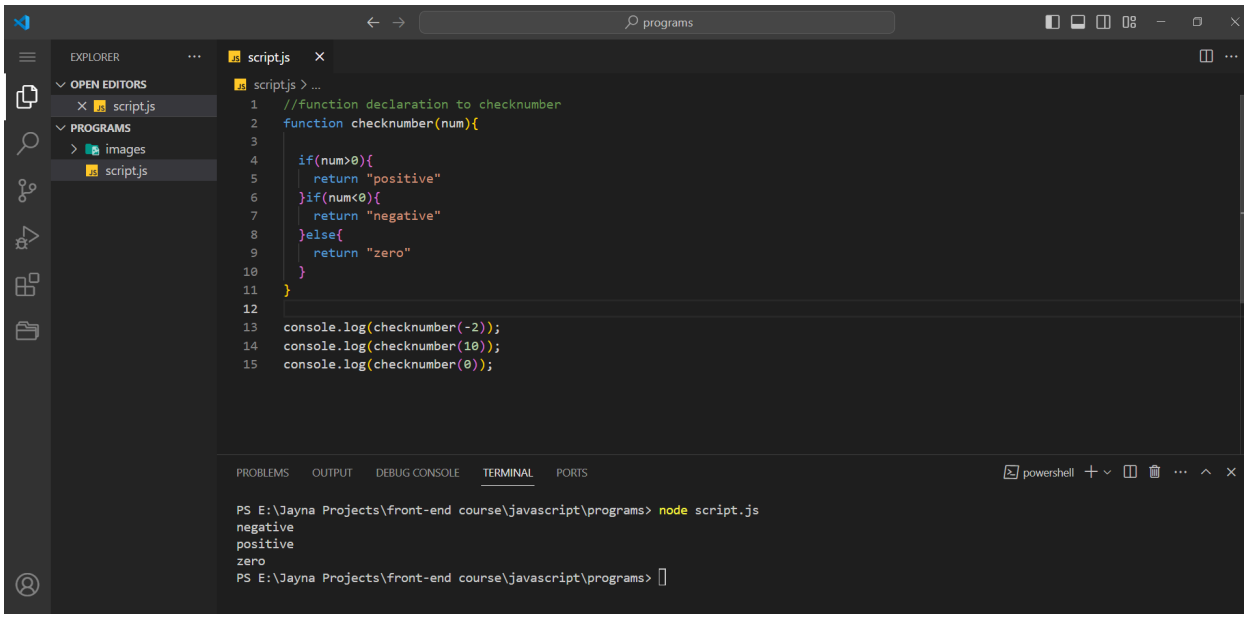
Q.22 Check Number Is Positive or Negative in JavaScript?



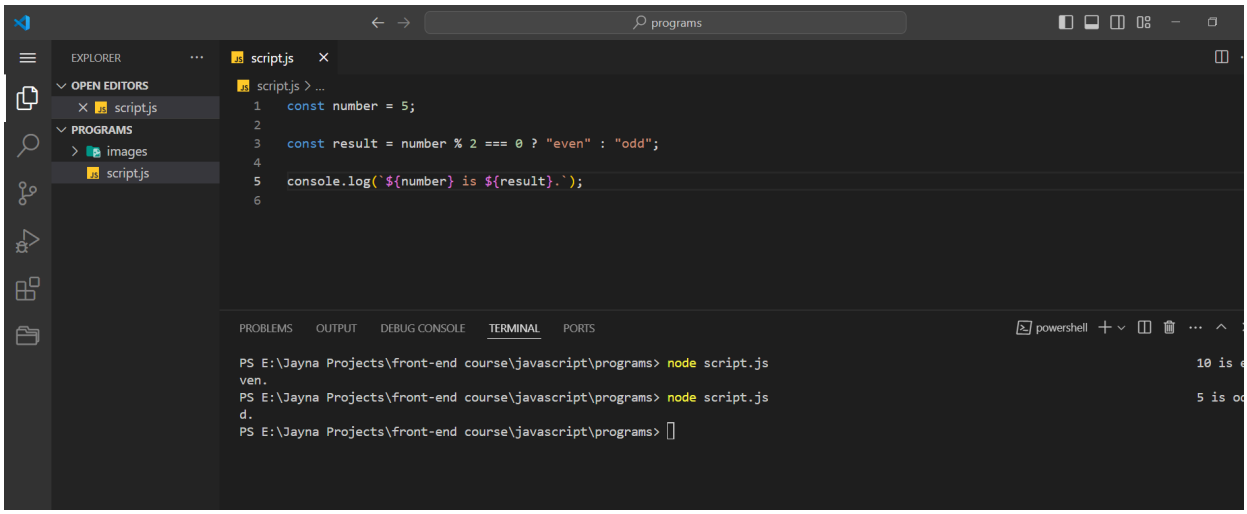
Q.23 Find the Character Is Vowel or Not ?



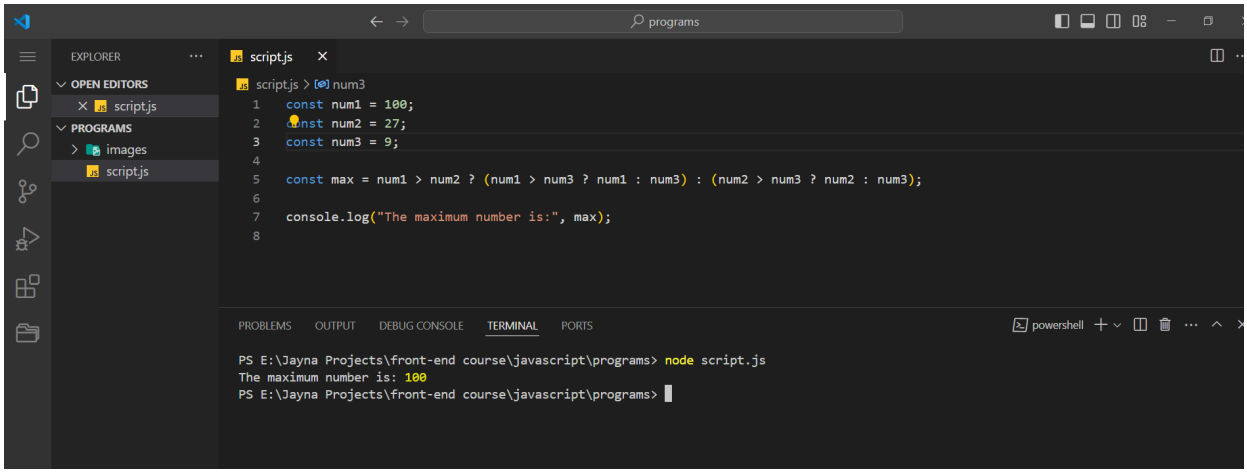
Q.24 Write to check whether a number is negative, positive or zero?



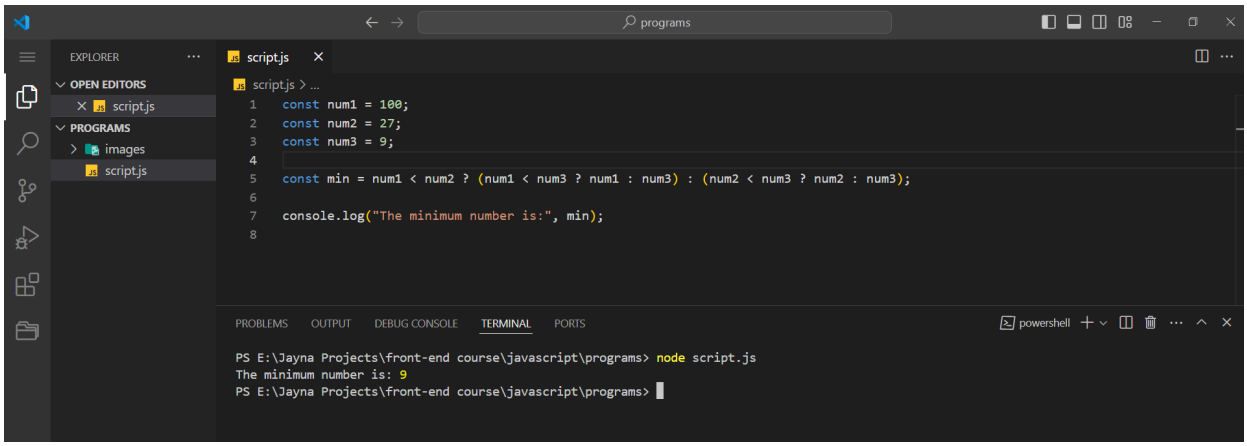
Q.25 Write to find number is even or odd using ternary operator in JS?



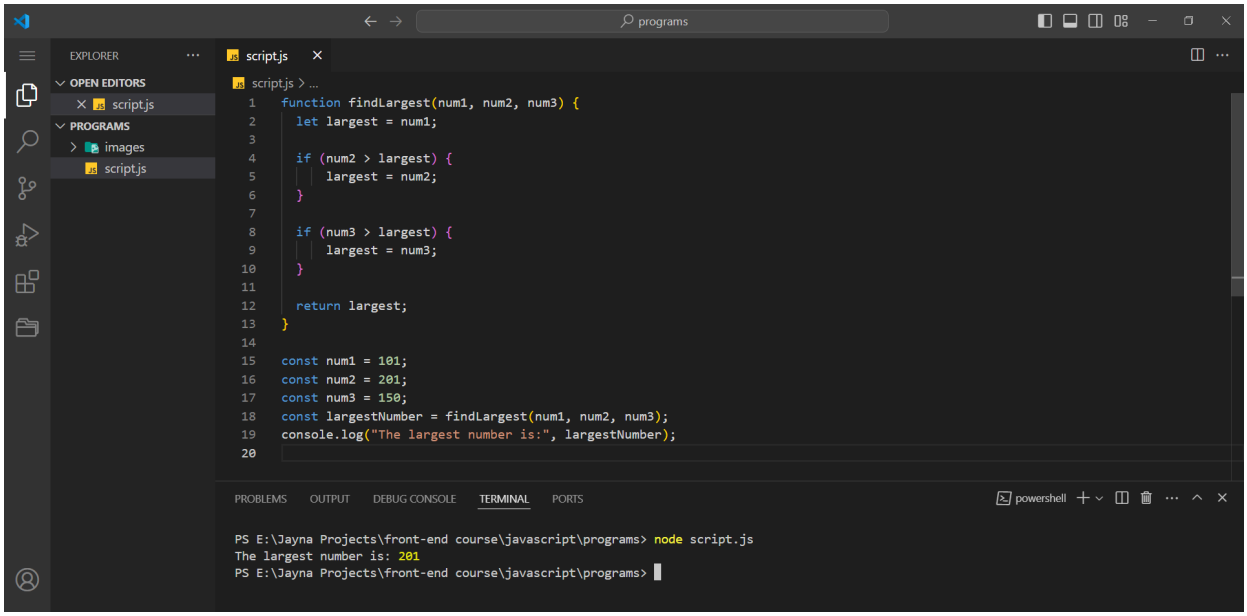
Q.26 Write find maximum number among 3 numbers using ternary operator in JS?



Q.27 Write to find minimum number among 3 numbers using ternary operator in JS?

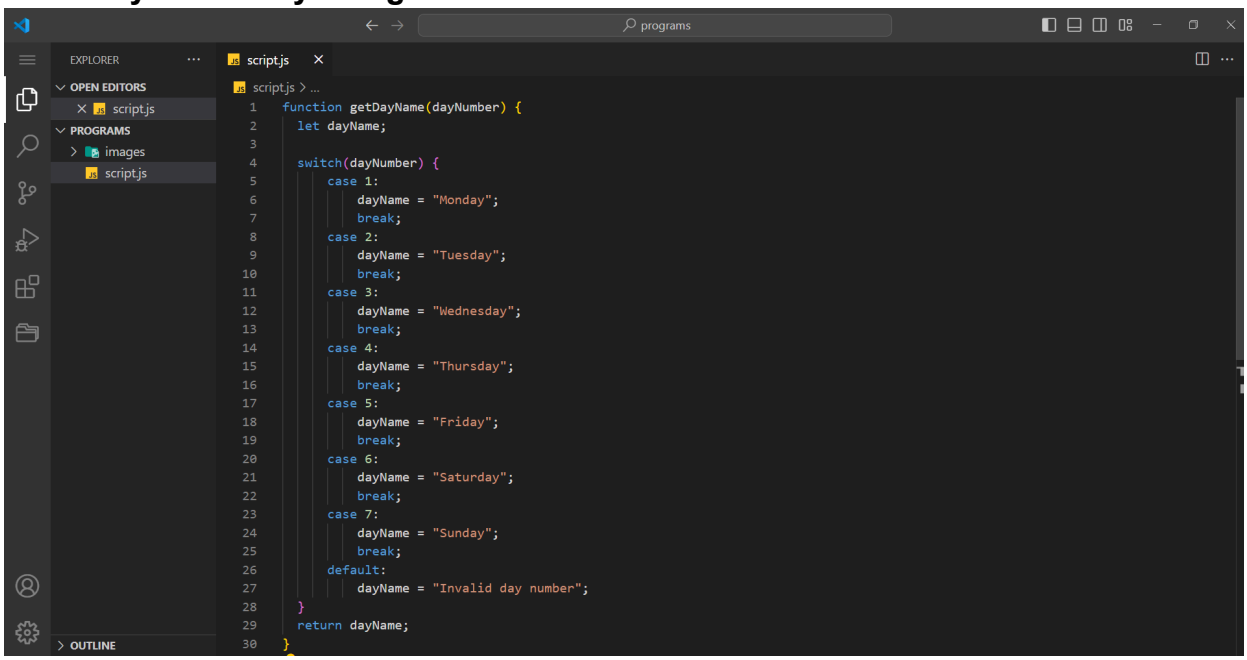


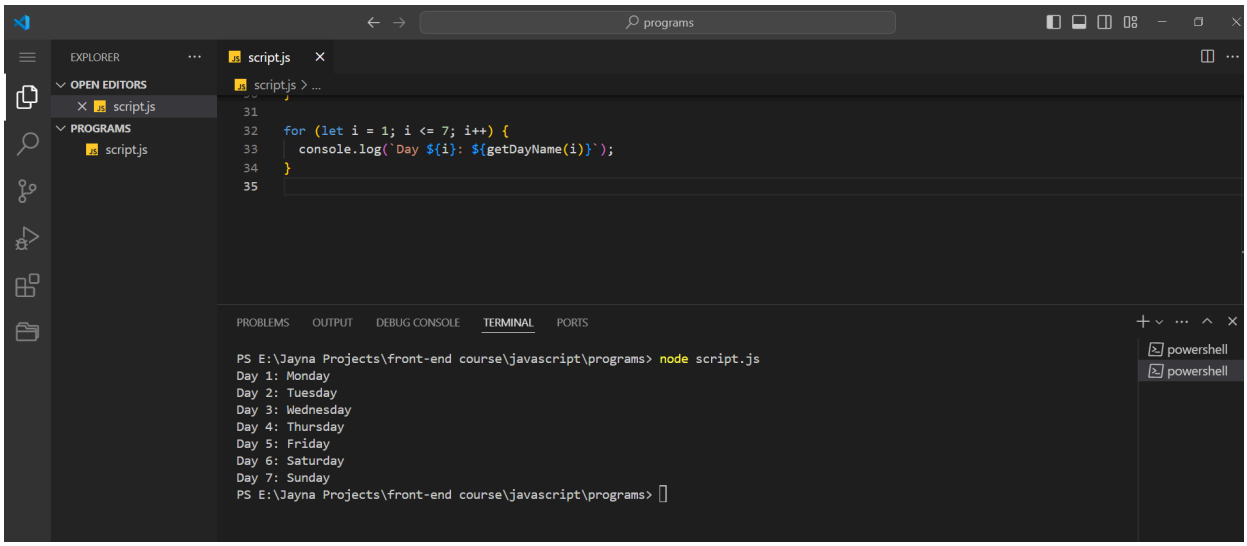
Q.28 Write to find the largest of three numbers in JS?



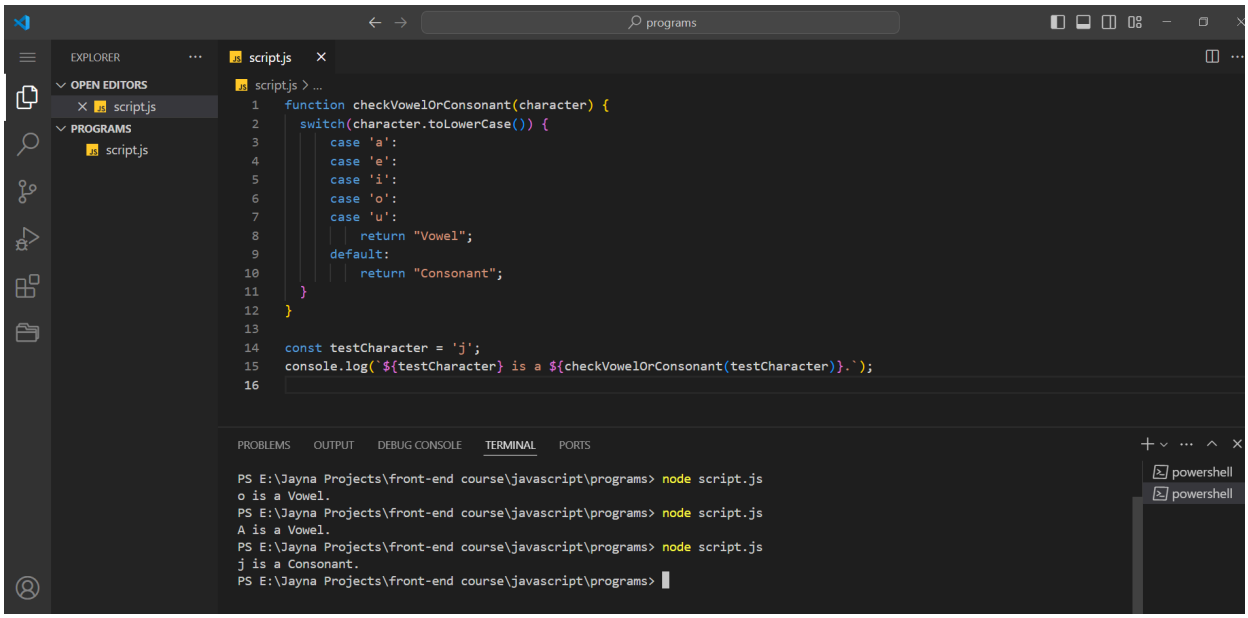
Q.29 Write a program to show :

i. Monday to Sunday using switch case in JS?





ii. Vowel or Consonant using switch case in JS?



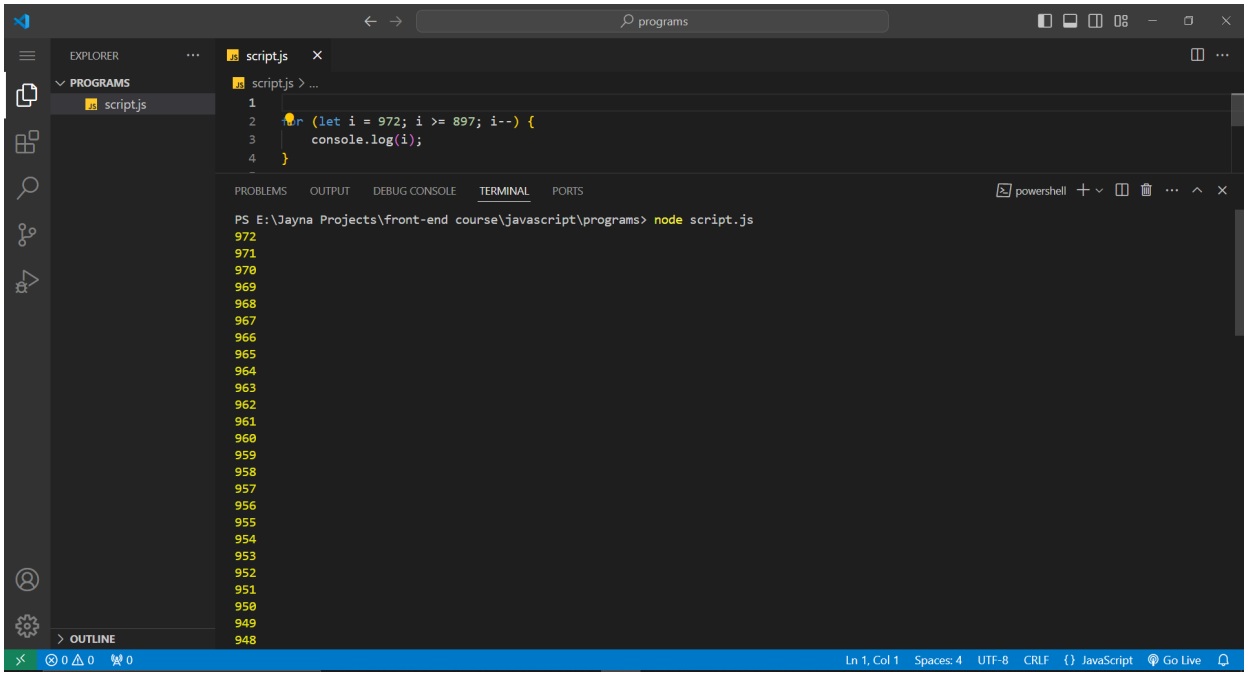
(Conditional looping logic Question)

Q.30 What are the looping structures in JavaScript? Any one Example?

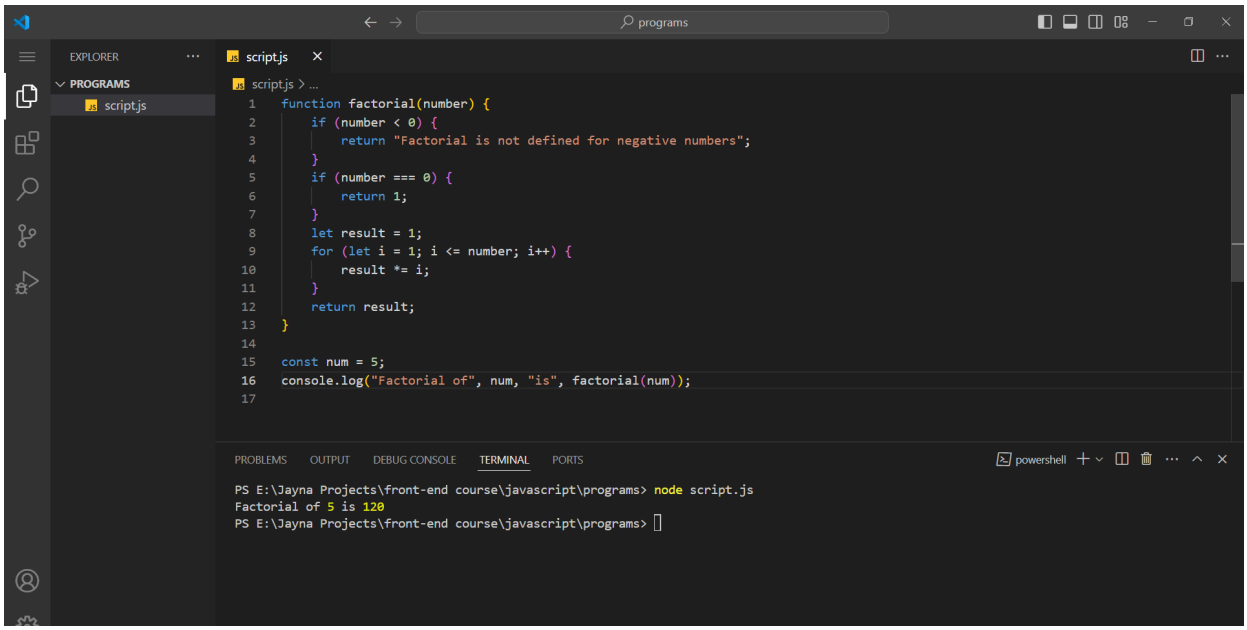
- JavaScript Loops are powerful tools for performing repetitive tasks efficiently. Loops in JavaScript execute a block of code again and again while the condition is true.
- Looping structure in javascript are:
  for - loops through a block of code a number of times
  for/in - loops through the properties of an object
  for/of - loops through the values of an iterable object
  while - loops through a block of code while a specified condition is true
  do while - also loops through a block of code while a specified condition is true

- Example:
  for (let i = 0; i < 5; i++) {
  console.log(i);
  }

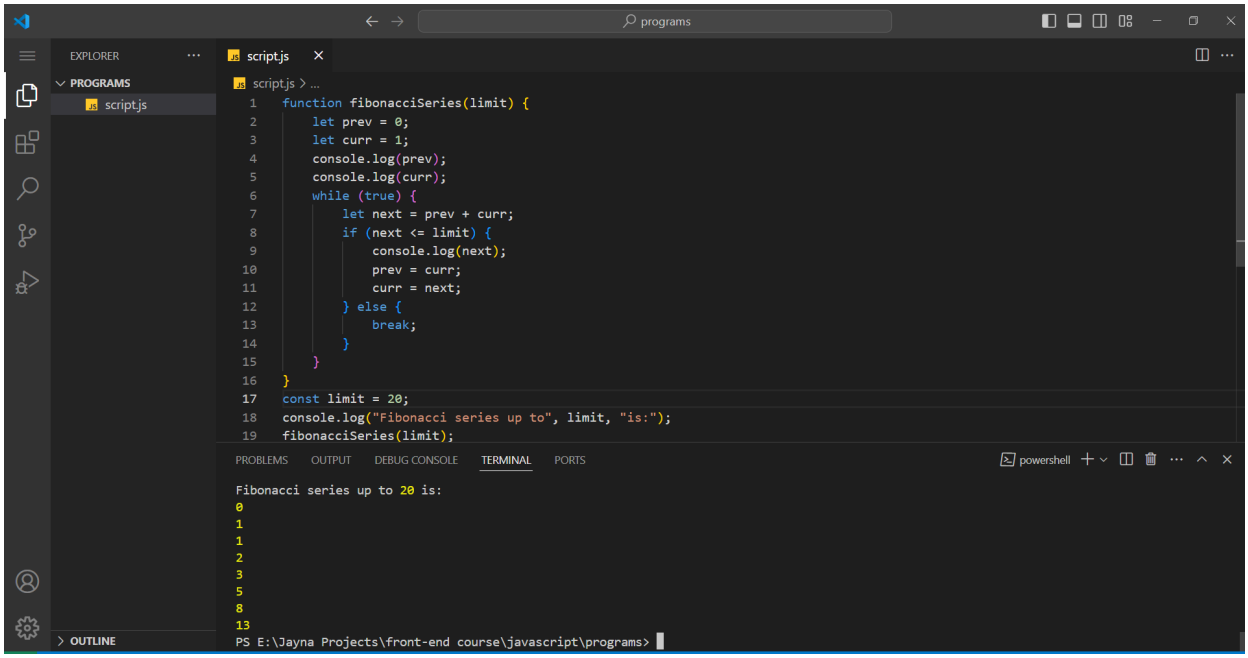
Q.31 Write a print 972 to 897 using for loop in JS?



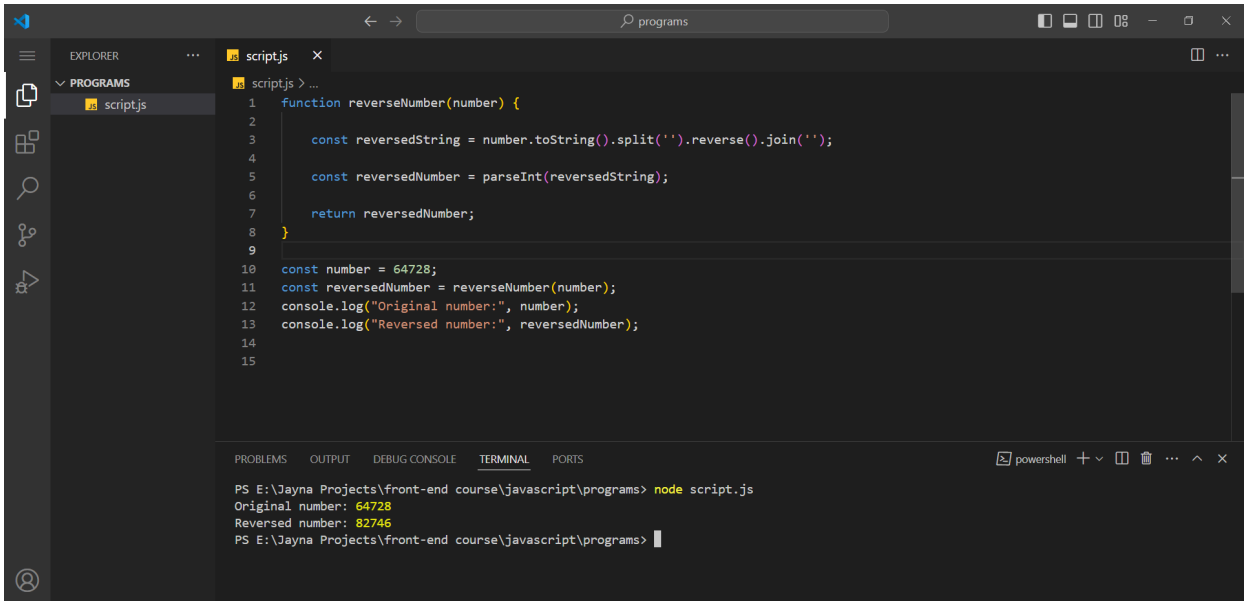
Q.32 Write to print the factorial of a given number?



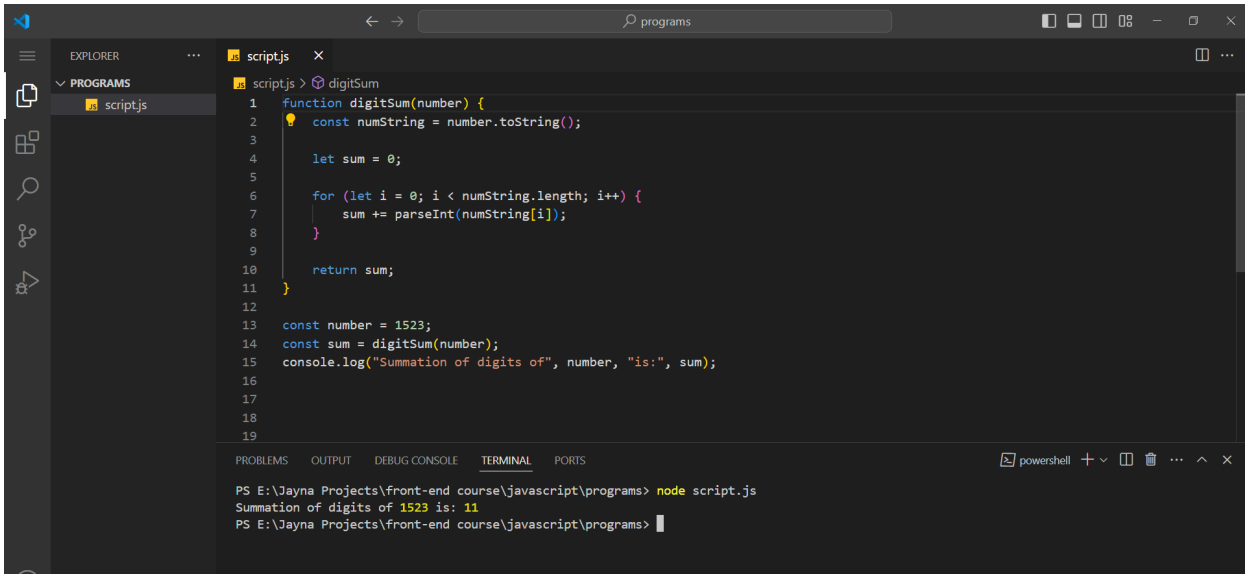
Q.33 Write to print Fibonacci series up to given numbers?



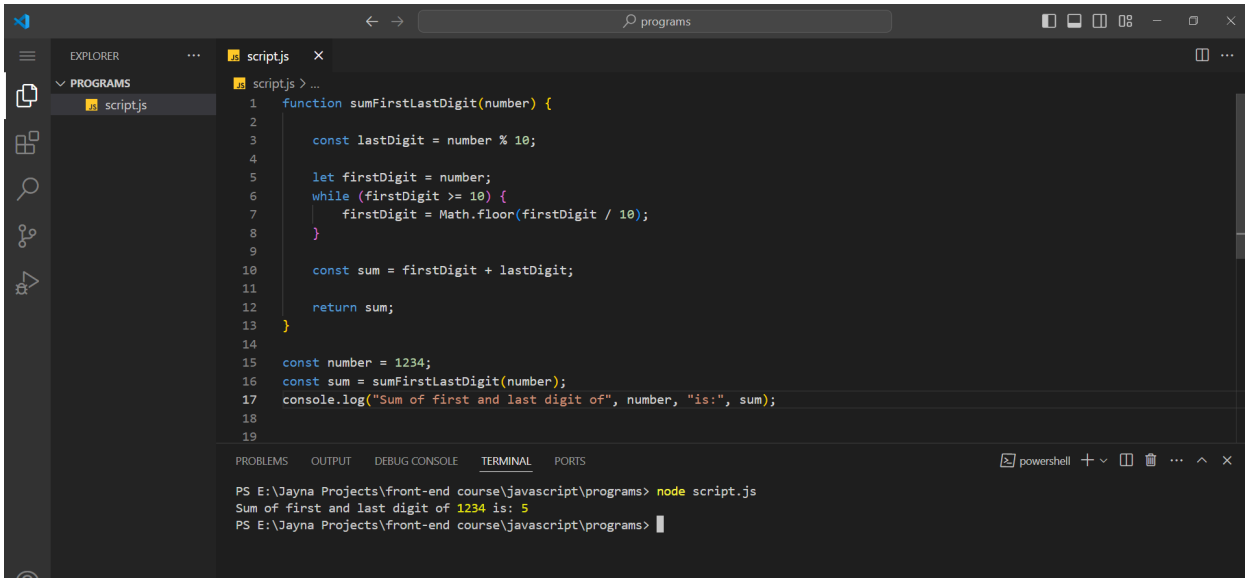
Q.34 Write to print number in reverse order e.g.: number = 64728 ---> reverse =82746 in JS?



Q.35 Write a program make a summation of given number (E.g., 1523 Ans: - 11) in JS?

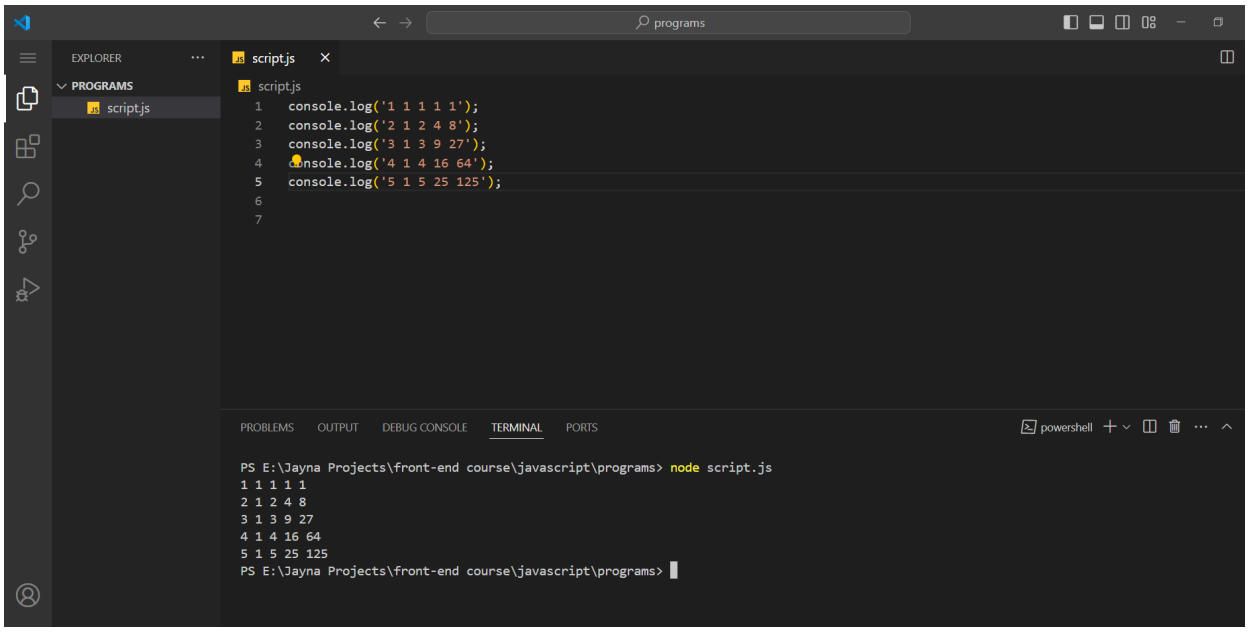


Q.36 Write a program you have to make a summation of first and last Digit. (E.g., 1234 Ans: - 5) in JS?



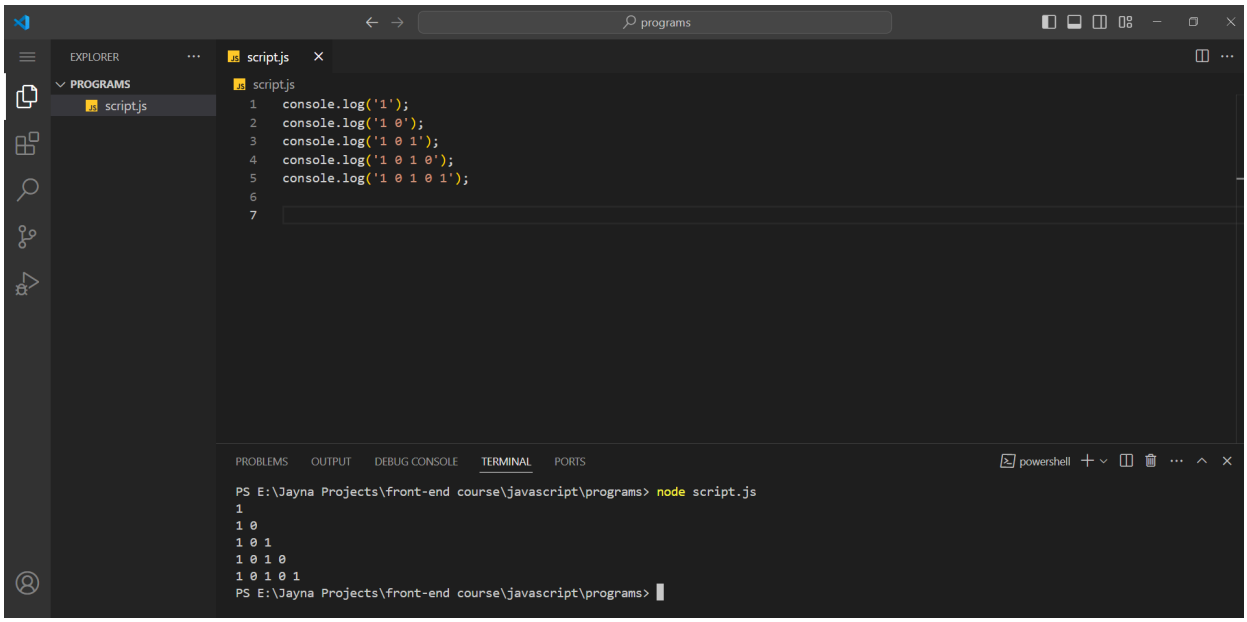
Q.37 Use console.log() and escape characters to print the following pattern in JS?

1 1 1 1 1
2 1 2 4 8
3 1 3 9 27
4 1 4 16 64
5 1 5 25 125



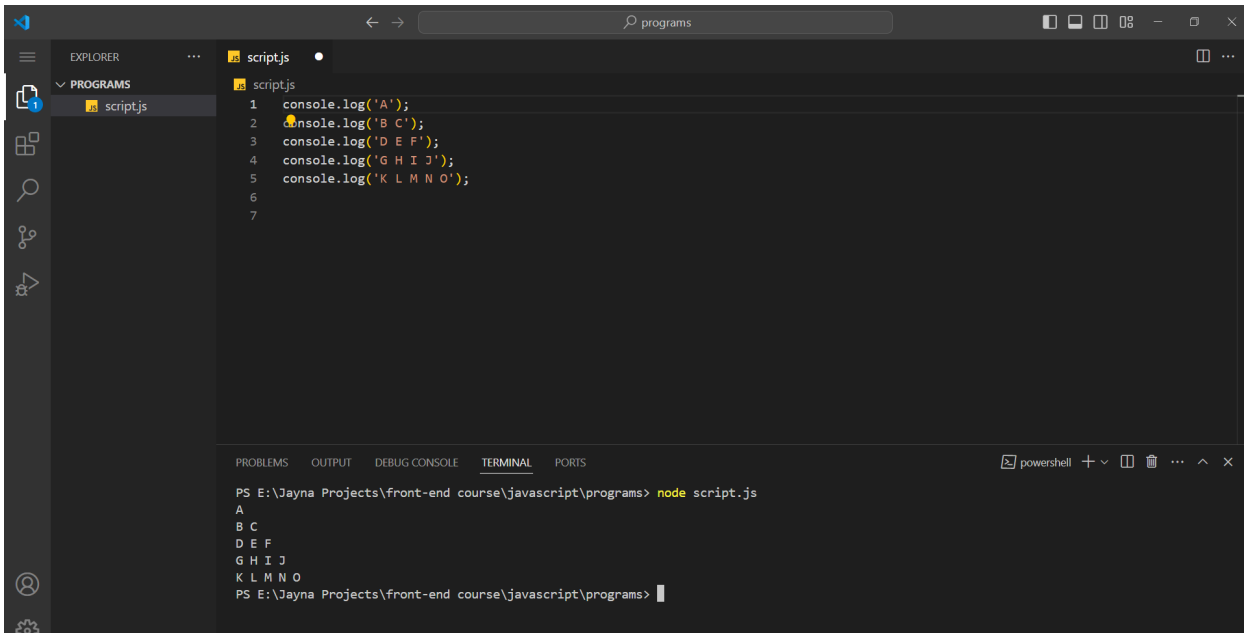
Q.38 Use pattern in console.log in JS?

1) 1
1 0
1 0 1
1 0 1 0
1 0 1 0 1

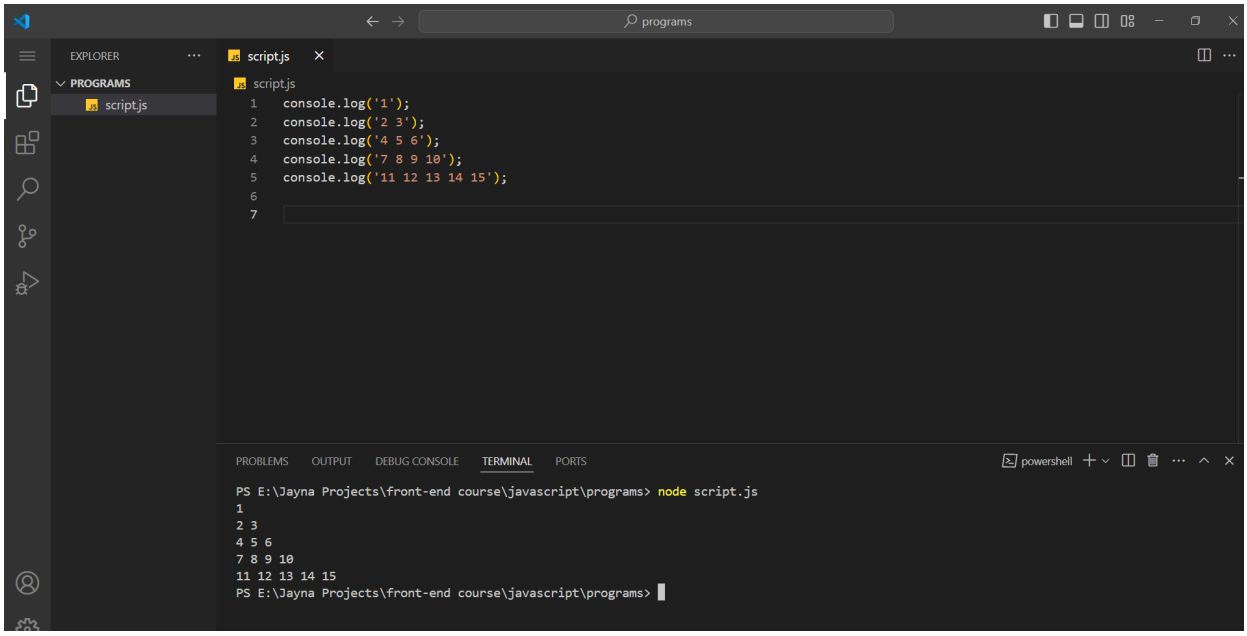


2) A
B C
D E F

G H I J
K L M N O



3) 1
2 3
4 5 6
7 8 9 10
11 12 13 14 15

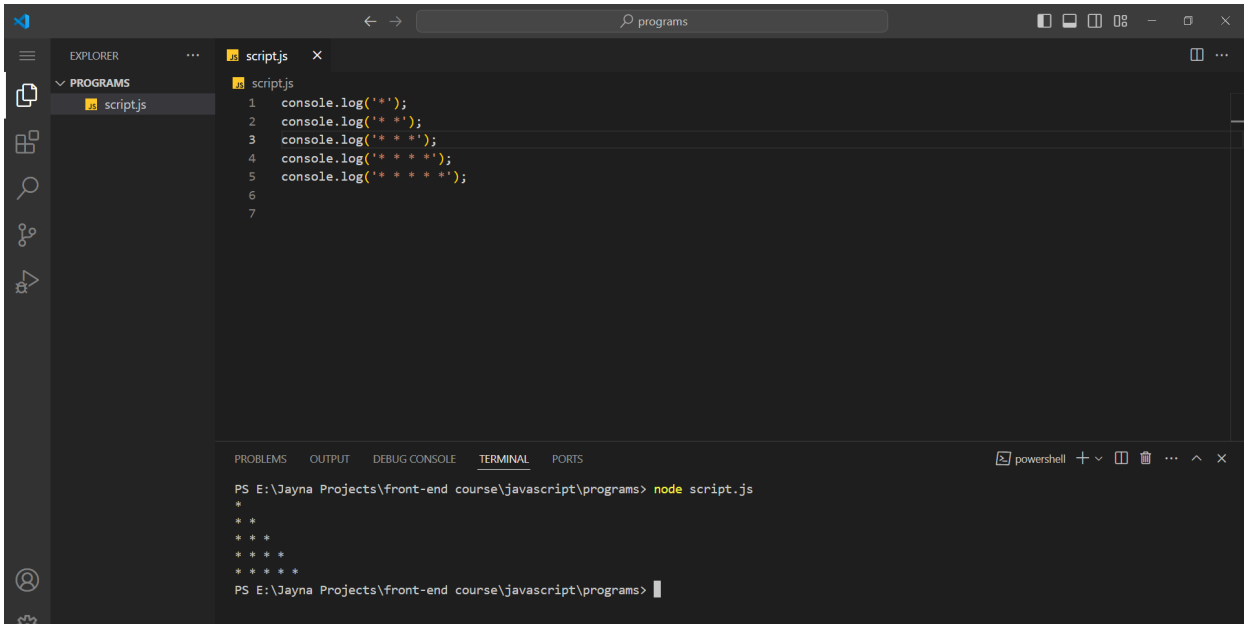


4) *
*** ***

* * *

* * * *

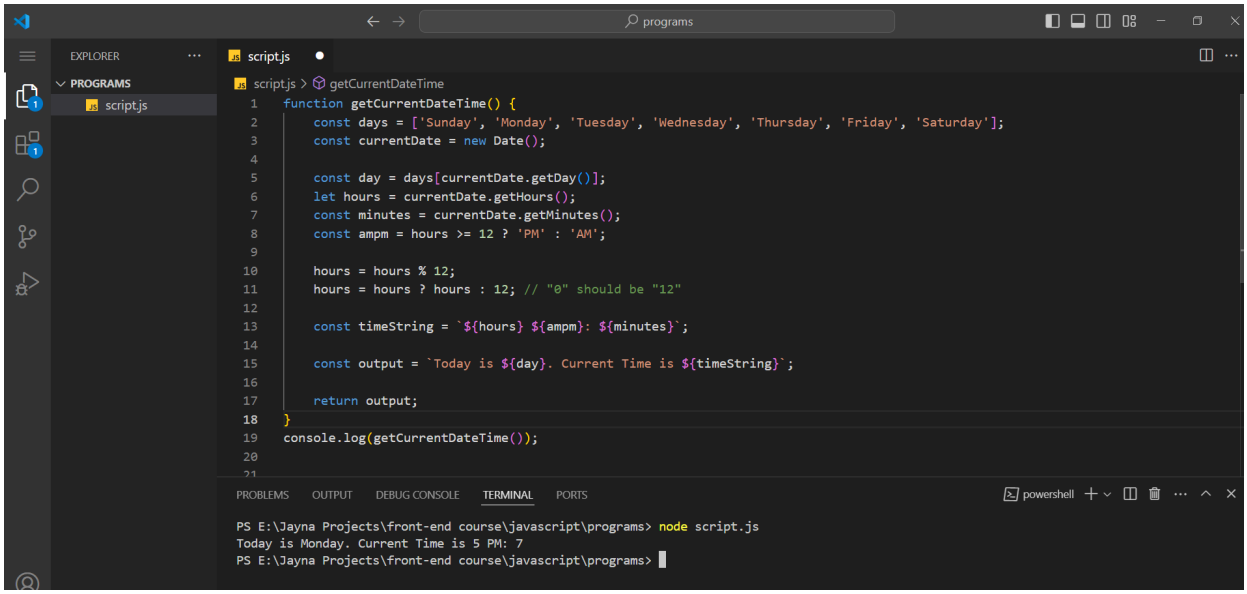
* * * * *



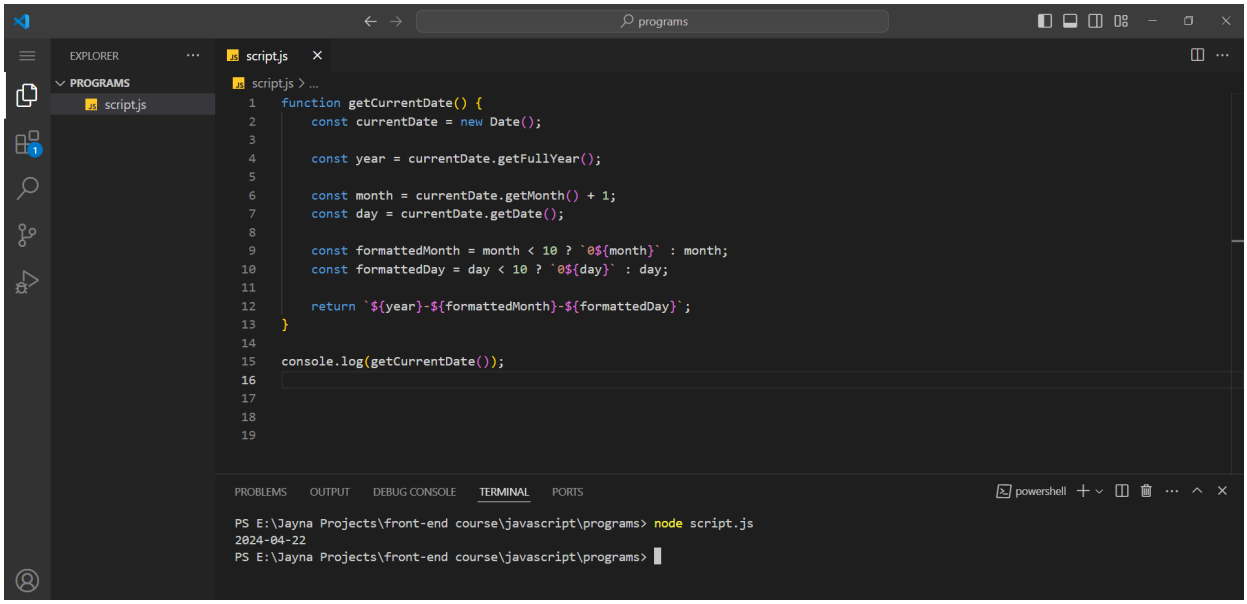
Q.39 Accept 3 numbers from user using while loop and check each numbers palindrome?

(Array and object Question)

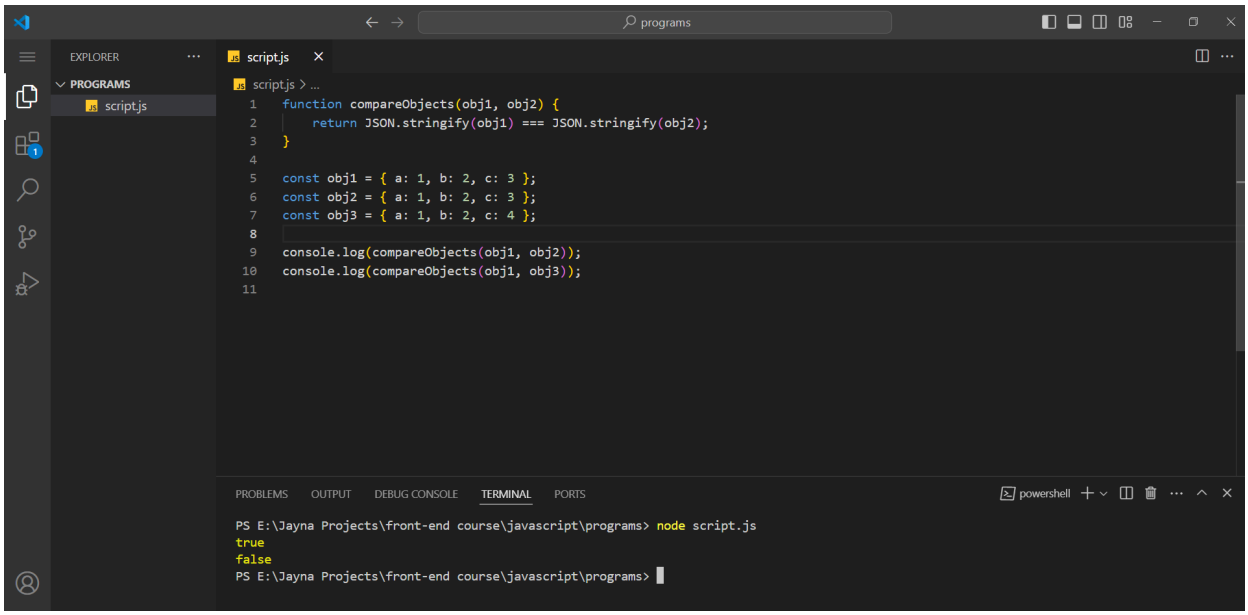
Q.40 Write a JavaScript Program to display the current day and time in the following format. Sample Output: Today is Friday. Current Time is 12 PM: 12 : 22 2 ?



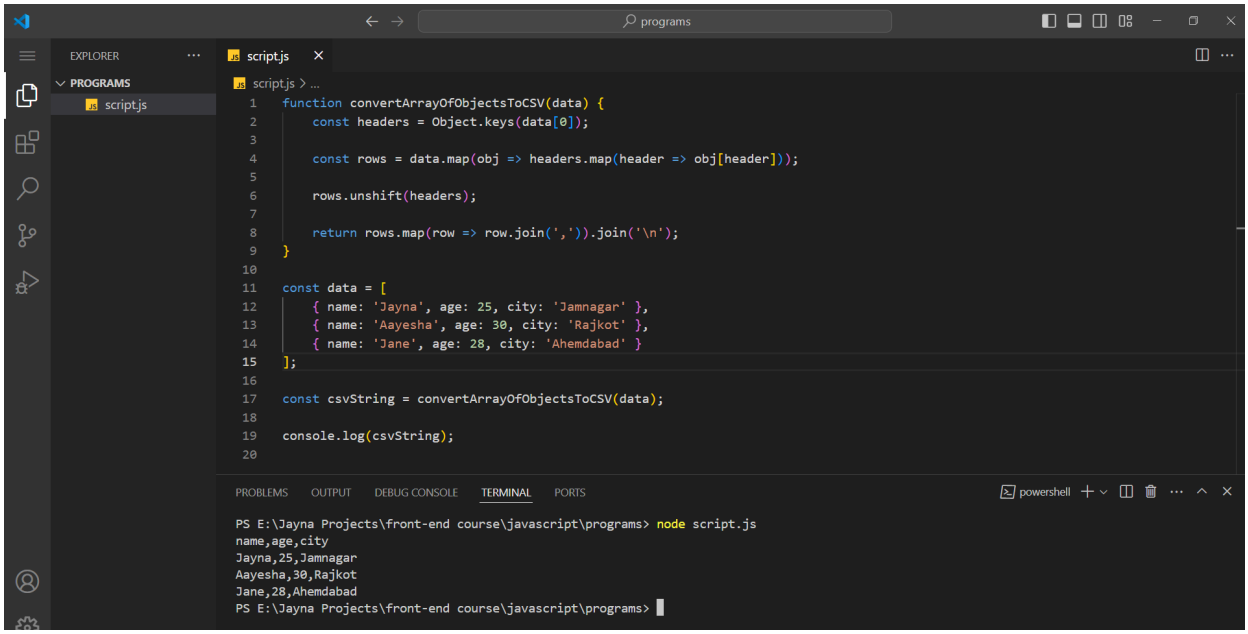
Q.41 Write a JavaScript program to get the current date?



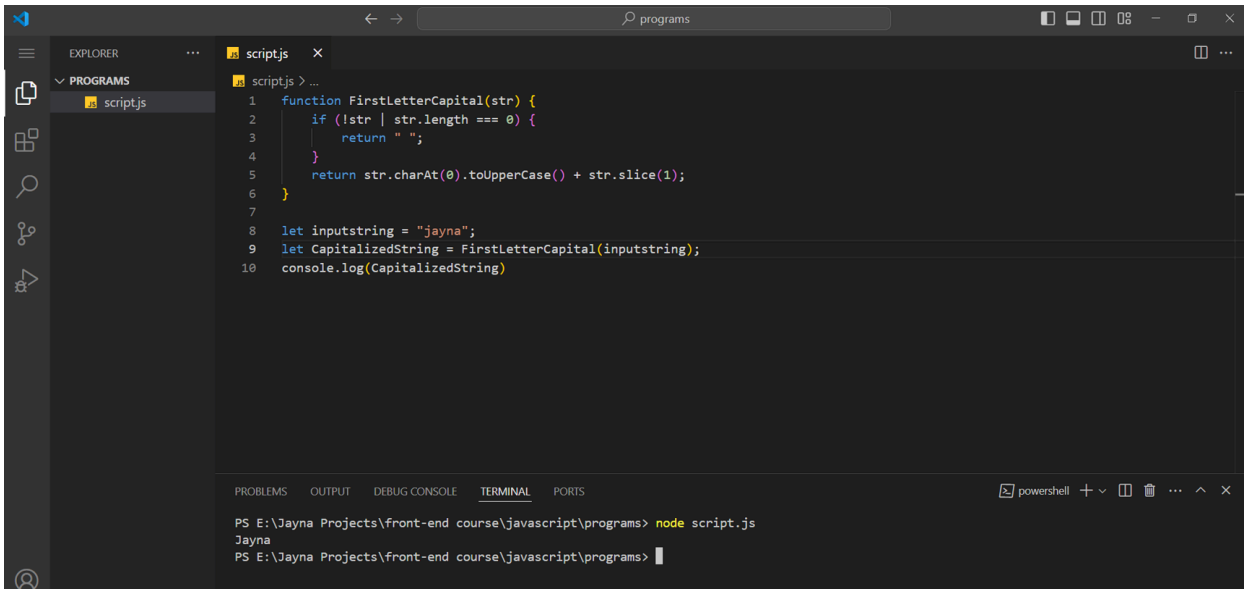
Q.42 Write a JavaScript program to compare two objects?



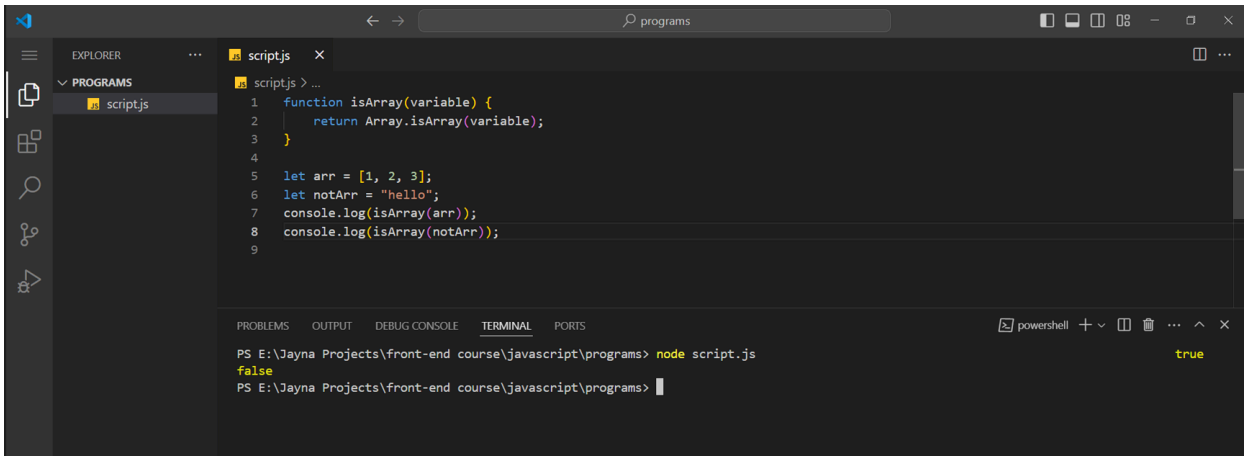
Q.43 Write a JavaScript program to convert an array of objects into CSV string?



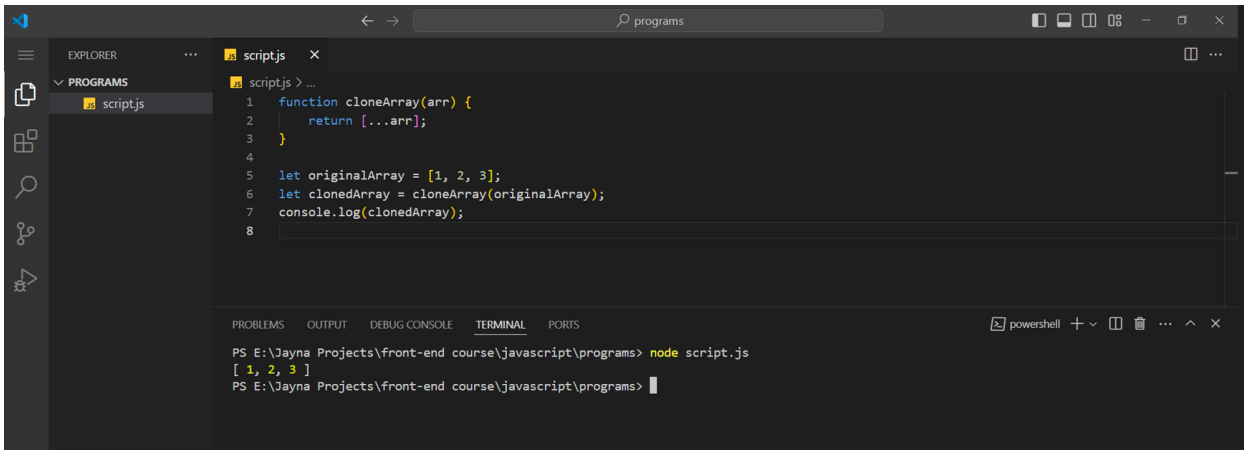
Q.44 Write a JavaScript program to capitalize the first letter of a string?



Q.45 Write a JavaScript program to determine if a variable is array?



Q.46 Write a JavaScript program to clone an array?



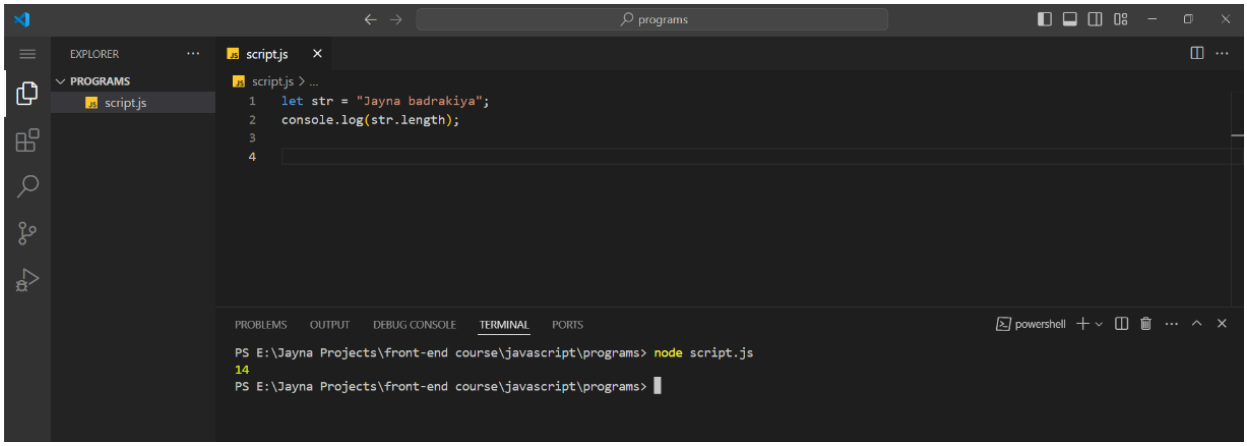
Q.47 What is the drawback of declaring methods directly in JavaScript objects?

- Declaring methods directly in JavaScript objects can lead to a drawback related to memory usage and performance, especially if those methods are intended to be shared among multiple instances of the object.

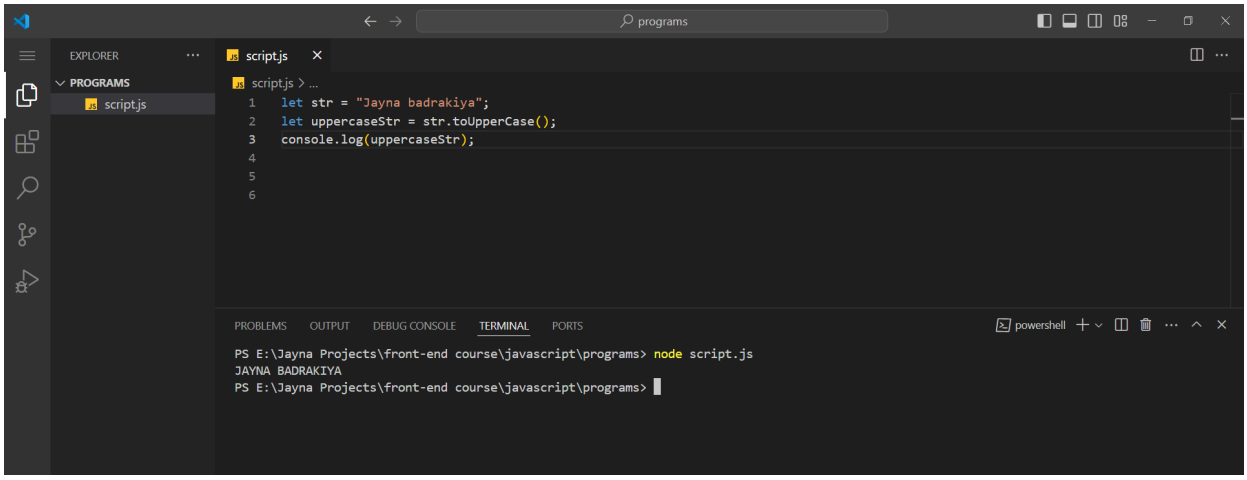
- Example:
  let myObject = {
  method()
  { }
  };

- Every time you create a new instance of ‘myobject’, JavaScript will allocate memory for that method separately for each instance. This can be inefficient if the method is the same for all instances and doesn't need to be recreated every time.

Q.48 Print the length of the string on the browser console using console.log()?



Q.49 Change all the string characters to capital letters using toUpperCase() method?



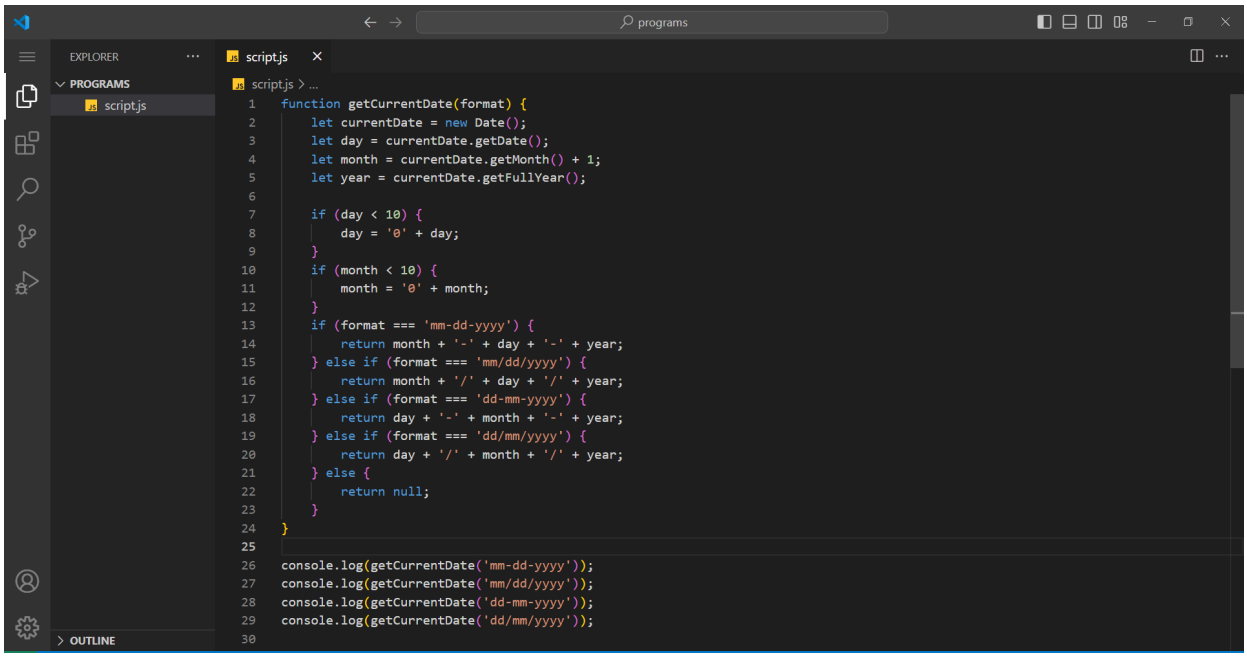
Q.50 What is the drawback of declaring methods directly in JavaScript objects?

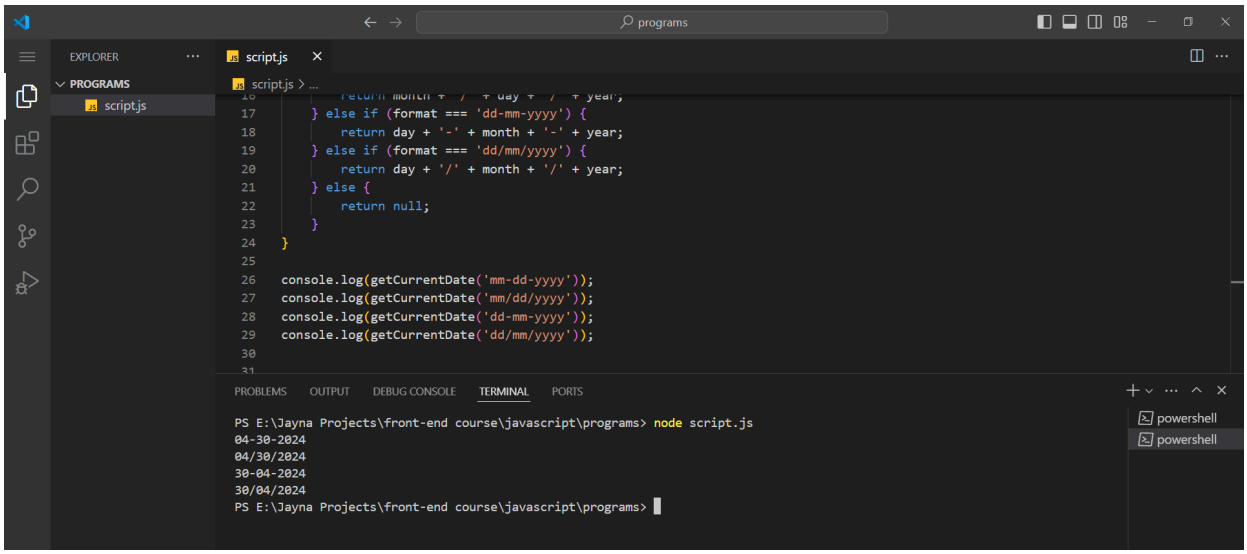
- Declaring methods directly in JavaScript objects can lead to a drawback related to memory usage and performance, especially if those methods are intended to be shared among multiple instances of the object.

- Example:
  let myObject = {
  method()
  { }
  };

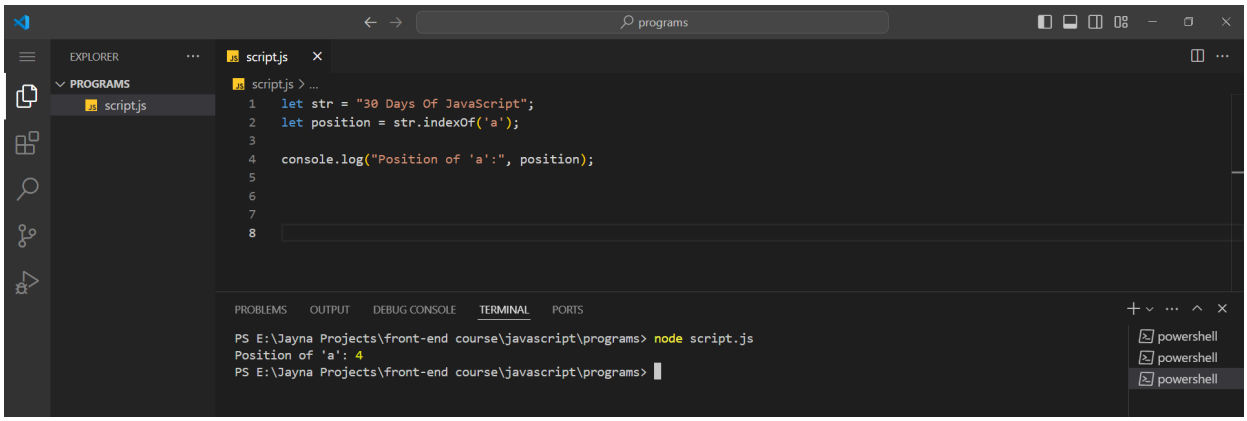
- Every time you create a new instance of ‘myobject’, JavaScript will allocate memory for that method separately for each instance. This can be inefficient if the method is the same for all instances and doesn't need to be recreated every time.

Q.51 Write a JavaScript program to get the current date. Expected Output : mm-dd-yyyy, mm/dd/yyyy or dd-mm-yyyy, dd/mm/yyyy?

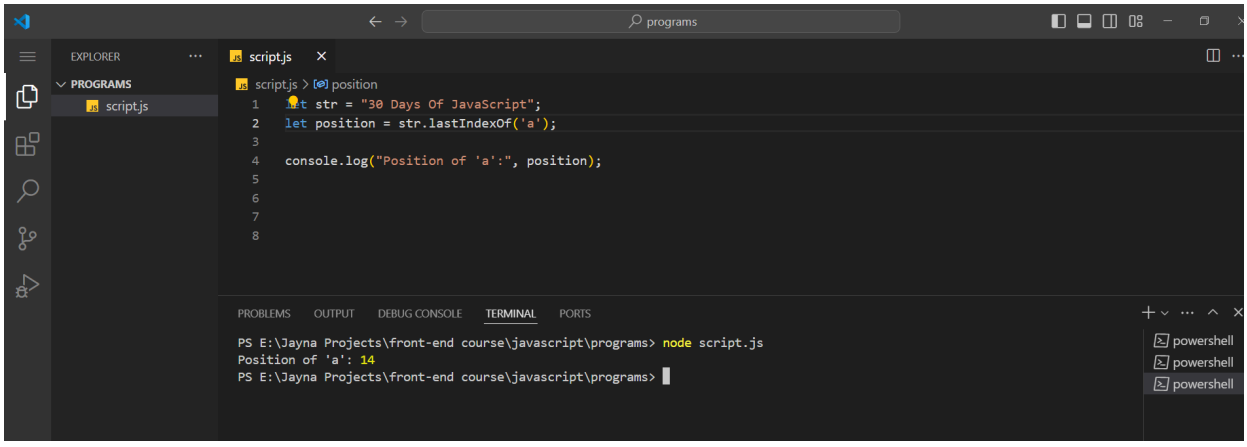




Q.52 Use indexOf to determine the position of the first occurrence of a in 30 Days Of JavaScript?



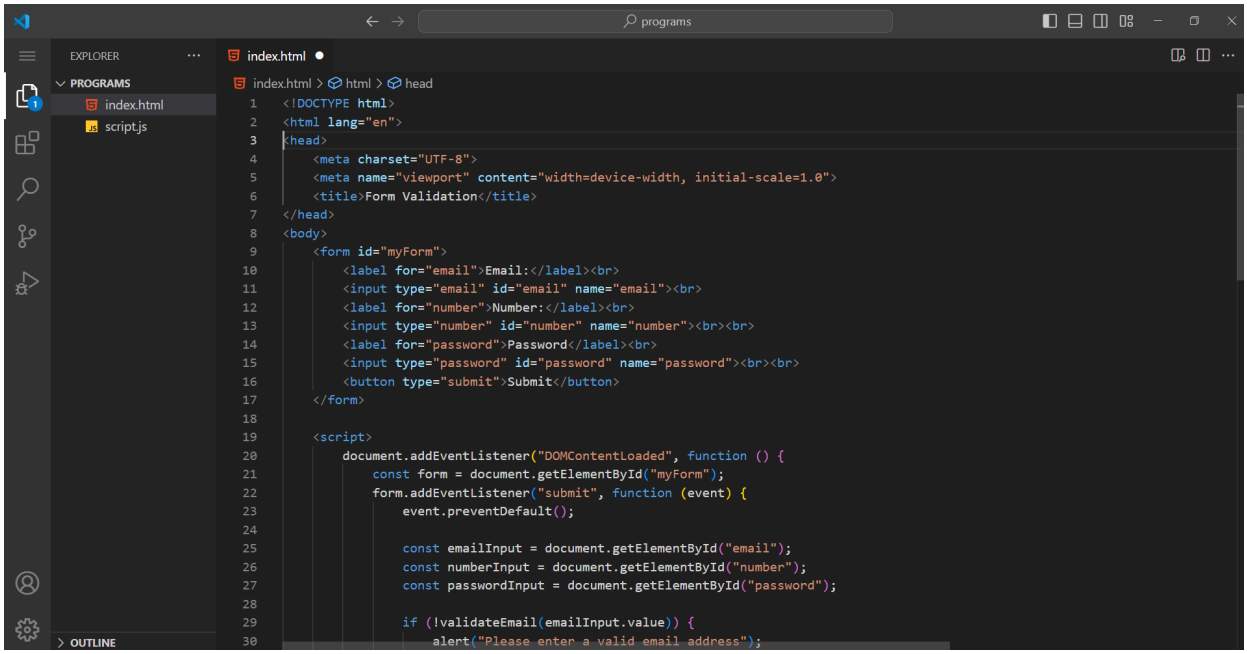
Q.53 Use lastIndexOf to determine the position of the last occurrence of a in 30 Days Of JavaScript?



Q.54 Form Validation in JS?

- Form validation in JavaScript involves checking the data entered into form fields to ensure it meets certain criteria or constraints before it is submitted to the server. This helps in ensuring that the data submitted is accurate, complete, and secure.

Q.55 Form in Email, number, Password, Validation?

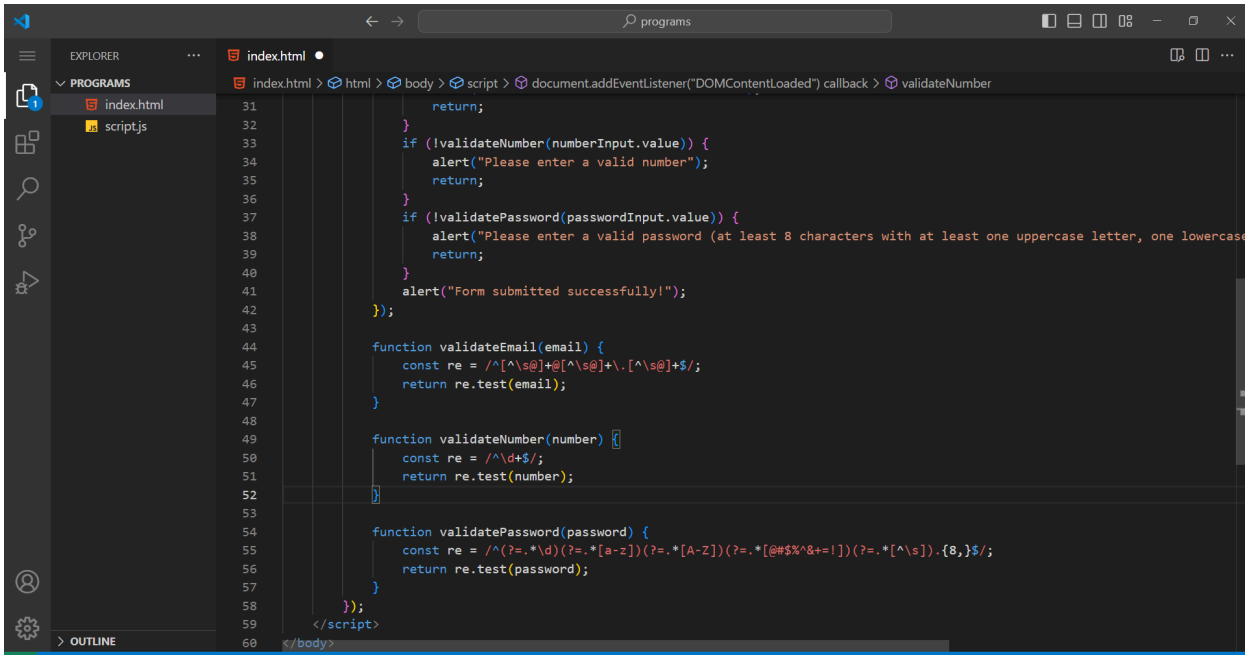


```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Form Validation</title>
</head>
<body>
    <form id="myForm">
        <label for="email">Email:</label><br>
        <input type="email" id="email" name="email"><br>
        <label for="number">Number:</label><br>
        <input type="number" id="number" name="number"><br><br>
        <label for="password">Password</label><br>
        <input type="password" id="password" name="password"><br><br>
        <button type="submit">Submit</button>
    </form>

    <script>
        document.addEventListener("DOMContentLoaded", function () {
            const form = document.getElementById("myForm");
            form.addEventListener("submit", function (event) {
                event.preventDefault();

                const emailInput = document.getElementById("email");
                const numberInput = document.getElementById("number");
                const passwordInput = document.getElementById("password");

                if (!validateEmail(emailInput.value)) {
                    alert("Please enter a valid email address");
```

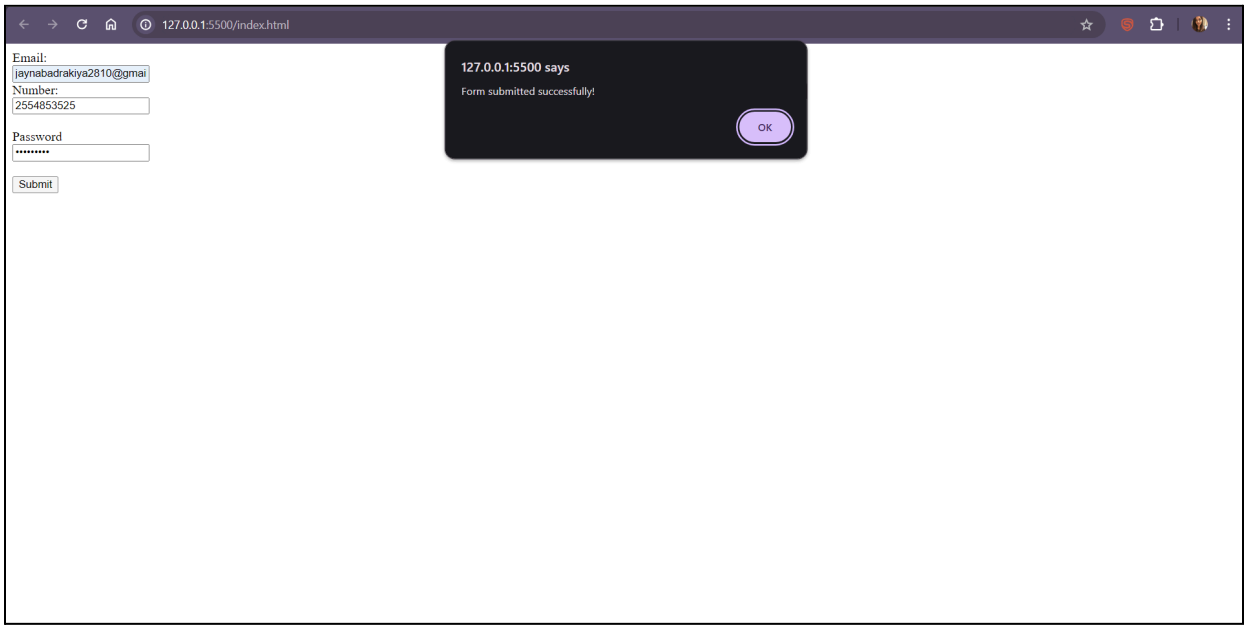


```
                    return;
                }
                if (!validateNumber(numberInput.value)) {
                    alert("Please enter a valid number");
                    return;
                }
                if (!validatePassword(passwordInput.value)) {
                    alert("Please enter a valid password (at least 8 characters with at least one uppercase letter, one lowercase
                    return;
                }
                alert("Form submitted successfully!");
            });

            function validateEmail(email) {
                const re = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
                return re.test(email);
            }

            function validateNumber(number) {
                const re = /^\d+$/;
                return re.test(number);
            }

            function validatePassword(password) {
                const re = /^(?=.*\d)(?=.*[a-z])(?=.*[A-Z])(?=.*[@#$%^&+=!])(?=.*[^\s]).{8,}$/;
                return re.test(password);
            }
        });
    </script>
</body>
```



Q.56 Dynamic Form Validation in JS?

- Dynamic form validation in JavaScript involves real-time validation of form inputs as the user interacts with the form. Unlike traditional validation where errors are displayed only upon form submission, dynamic validation provides immediate feedback to the user, often as they type or move focus away from an input field.

Q.57 How many types of JS Event? How to use it ?

- JavaScript events are actions or occurrences that happen in the browser, triggered by user interaction, browser actions, or by other JavaScript code. Events allow JavaScript to respond to user input or system actions, enabling interactive and dynamic web pages. There are many types of events in JavaScript. Here are some common types:

1) Mouse Events: These events are triggered by mouse actions.
   click: Triggered when the element is clicked.
   dblclick: Triggered when the element is double-clicked.
   mouseover: Triggered when the mouse pointer enters the element.
   mouseout: Triggered when the mouse pointer leaves the element.
   mousedown: Triggered when the mouse button is pressed down on the element.
   mouseup: Triggered when the mouse button is released over the element.
   mousemove: Triggered when the mouse pointer moves over the element.
   mouseenter: Similar to mouseover, but not triggered by event bubbling.

2) Keyboard Events: These events are triggered by keyboard actions.
   keydown: Triggered when a key is pressed down.
   keyup: Triggered when a key is released.
   keypress: Triggered when a key that produces a character value is pressed down.

3) Form Events: These events are triggered by form elements.
   submit: Triggered when a form is submitted.

reset: Triggered when a form is reset.
change: Triggered when the value of a form element changes (e.g., input field, select box).
input: Triggered when the value of an input field changes.
focus: Triggered when an element receives focus.
blur: Triggered when an element loses focus.

4) Window Events: These events are triggered by actions related to the window or document.
load: Triggered when the page/document is fully loaded.
unload: Triggered when the page/document is being unloaded (e.g., navigating to a new page).
resize: Triggered when the browser window is resized.
scroll: Triggered when scrolling occurs on the window or an element.
DOMContentLoaded: Triggered when the HTML document has been completely loaded and parsed.

5) Event Delegation: This is not a specific event type but a concept in JavaScript where a single event listener is attached to a parent element, and events from child elements are handled based on a selector. This is useful for handling events on dynamically added elements or a large number of elements efficiently.

Example:

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

<button id="myButton">Click me</button>

<script>
const button = document.getElementById("myButton");
button.addEventListener("click", function() {
   alert("Button clicked!");
});
</script>

</body>
</html>
```

Q.59 What is Bom vs Dom in JS?

- In JavaScript, "BOM" stands for Browser Object Model, while "DOM" stands for Document Object Model.

Browser object model

- The browser object model is a set of objects provided by the browser, which allows javascript to communicate with the browser. It includes objects such as window, navigator, screen, history and location. These objects provide methods and properties to interact with browser functionalities like controlling the browser window, navigating to different URLs, managing cookies, and so on. Bom doesn't have any standard defined by W3C, and its specific to each browser implementation.

Document object model

- The document object model is a representation of the structure of HTML or XML documents. It provides a way for scripts to dynamically access and manipulate the content, structure and style of documents. The DOM organizes the elements of a document as a hierarchical tree, where each node represents an element like , <p>,<img> or other entities like text or comments.
- It allows javascript to interact with the content of a web page, enabling tasks such as adding or removing elements, changing styles, handling events and so on. DOM is standardized by the W3C and is implemented uniformly across browsers.

Q.60 Array vs object defences in JS?

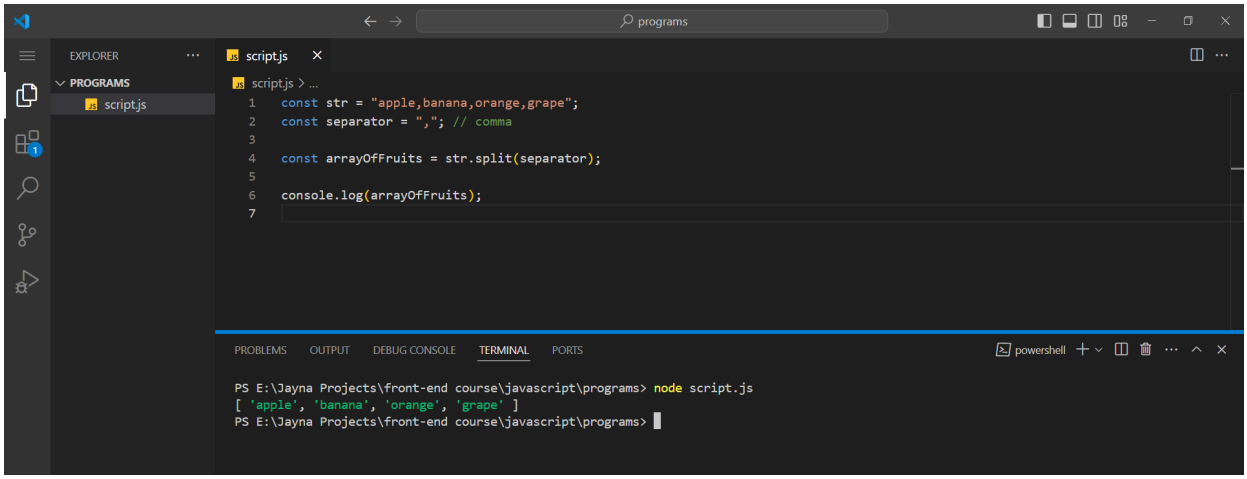
Array defences

- Boundary Checking: Always check array bounds to avoid accessing elements outside the array's range.
- Type Checking: Ensure that arrays contain elements of expected types to prevent unexpected behavior.
- Avoid Sparse Arrays: Sparse arrays (arrays with holes or gaps) can lead to unexpected behavior, so it's generally better to avoid them.
- Avoid Direct Modification of Array Prototype: Modifying the prototype of the Array object can lead to unexpected behavior. Avoid adding or removing properties from Array.prototype.

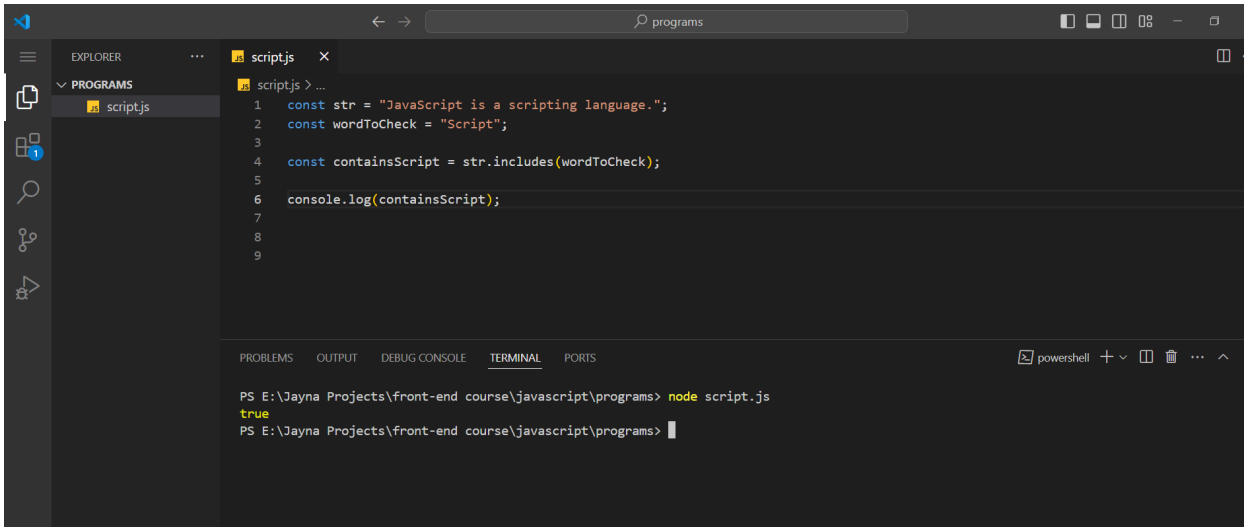
Object defences

- Property Existence Checking: Always check if a property exists before accessing it to avoid errors.
- Use hasOwnProperty: When iterating over object properties, use hasOwnProperty to make sure you're only iterating over the object's own properties, not inherited ones.
- Freezing Objects: Use Object.freeze() to make an object immutable, preventing properties from being added, modified, or removed.
- Sealing Objects: Use Object.seal() to prevent new properties from being added and existing ones from being deleted, but still allowing their values to be changed.
- Avoid Circular References: Be cautious of circular references in objects, as they can cause infinite loops when iterating over properties.

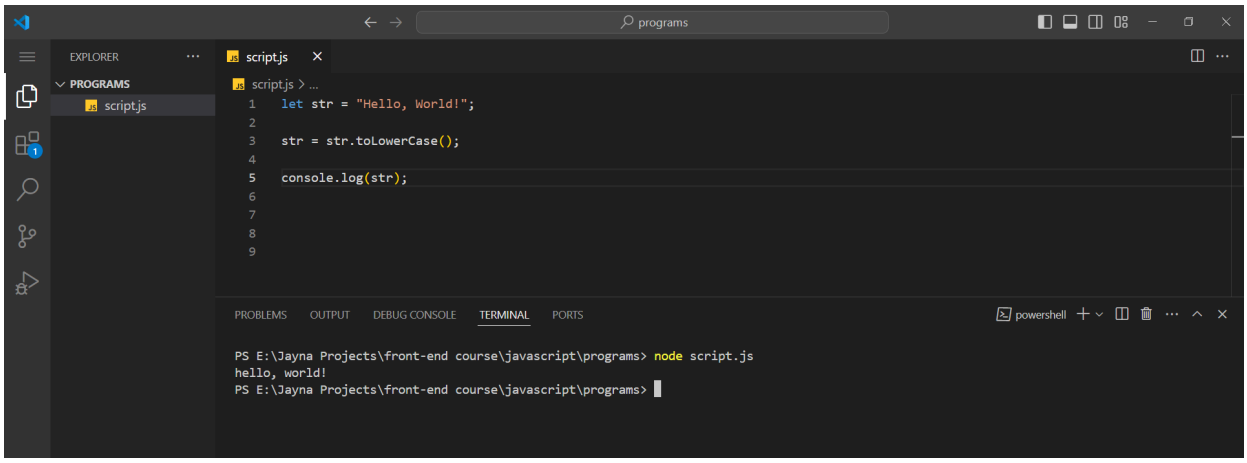
Q.61 Split the string into an array using split() Method?



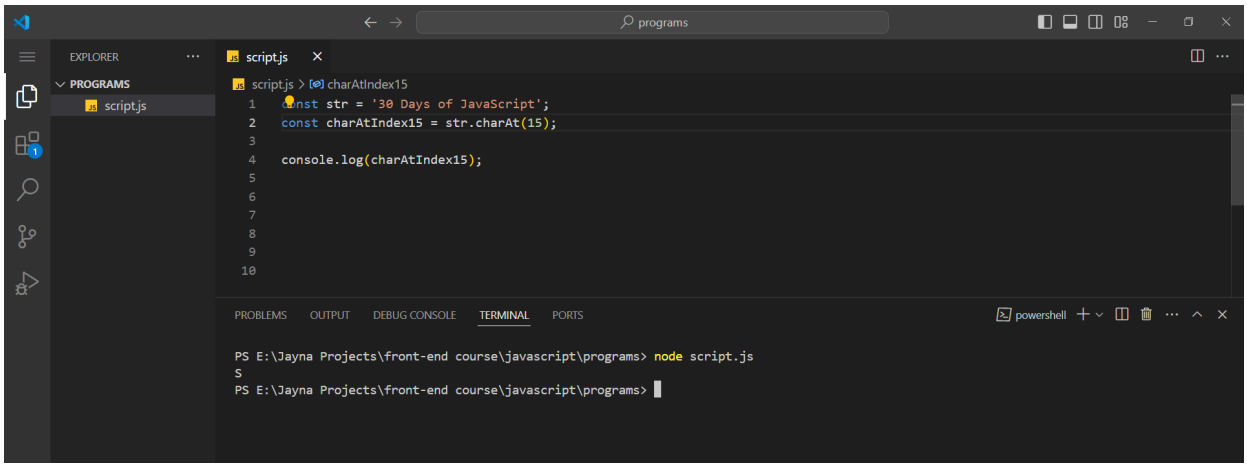
Q.62 Check if the string contains a word Script using includes() method?



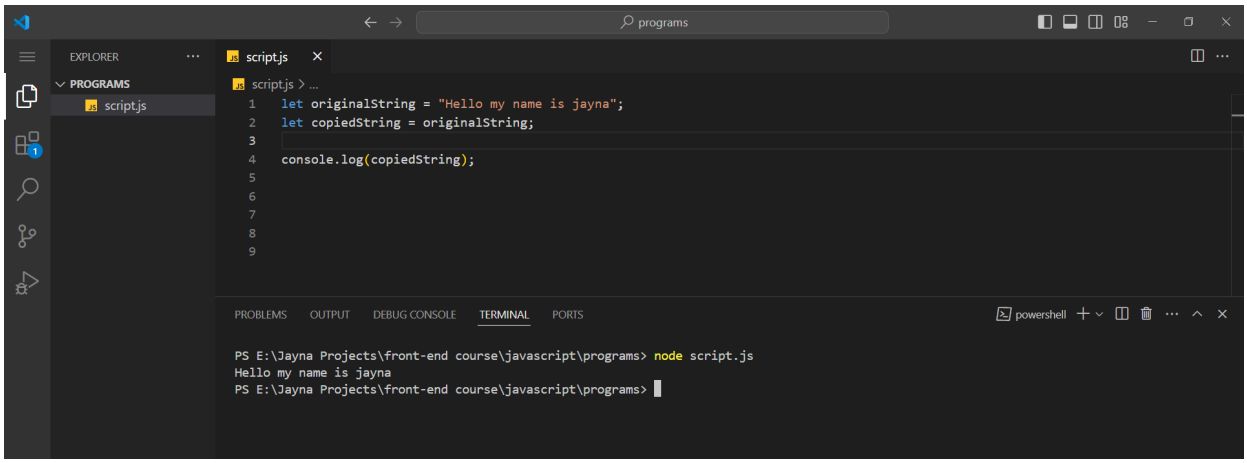
Q.63 Change all the string characters to lowercase letters using toLowerCase() Method.



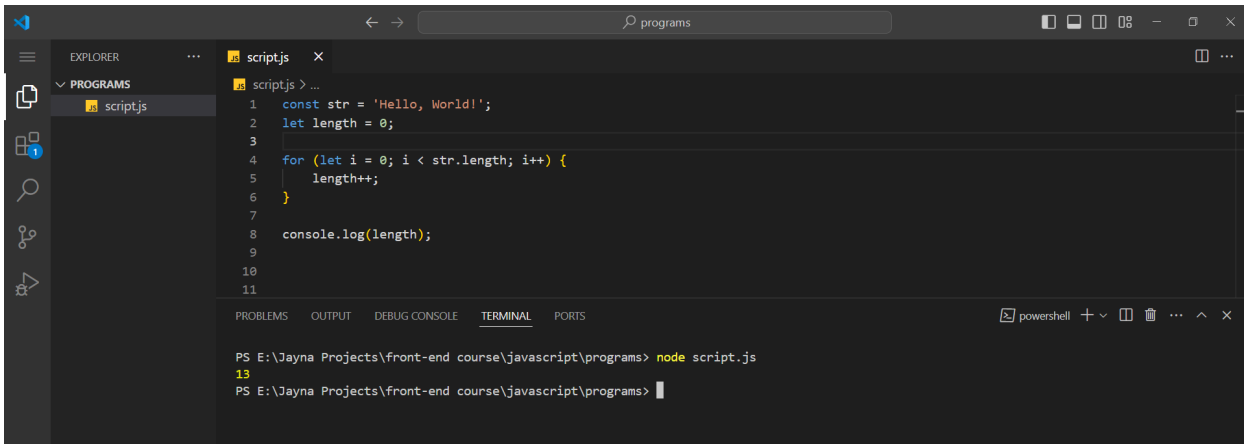
Q.64 What is Character at index 15 in '30 Days of JavaScript' string? Use charAt() method.



Q.65 copy to one string to another string in JS?



Q.66 Find the length of a string without using libraryFunction?



- **What is JavaScript?**
  - JavaScript is a lightweight, cross-platform, single-threaded, and interpreted compiled programming language. It is also known as the scripting language for web pages.
  - It is well-known for the development of web pages, and many non-browser environments also use it.
  - JavaScript can update and change both HTML and CSS.
  - JavaScript can calculate, manipulate and validate data.

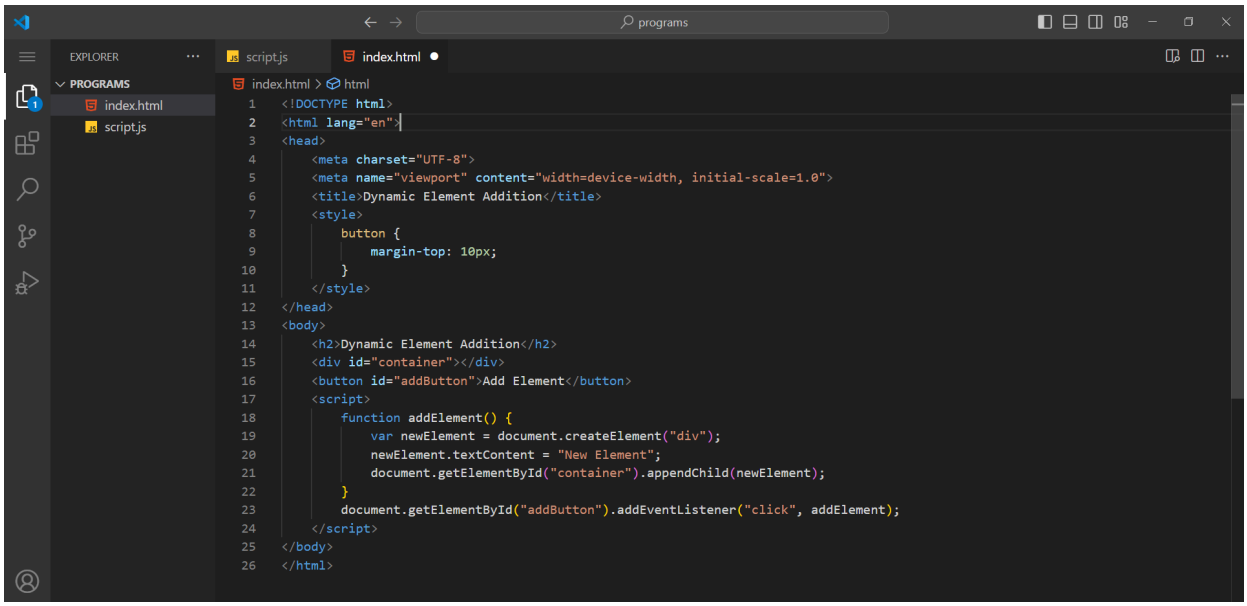
- **What is the use of isNaN function?**
  - In JavaScript, NaN is short for "Not-a-Number".
  - In JavaScript, NaN is a number that is not a legal number.
  - The Number.isNaN() method returns true if the value is NaN, and the type is a Number.

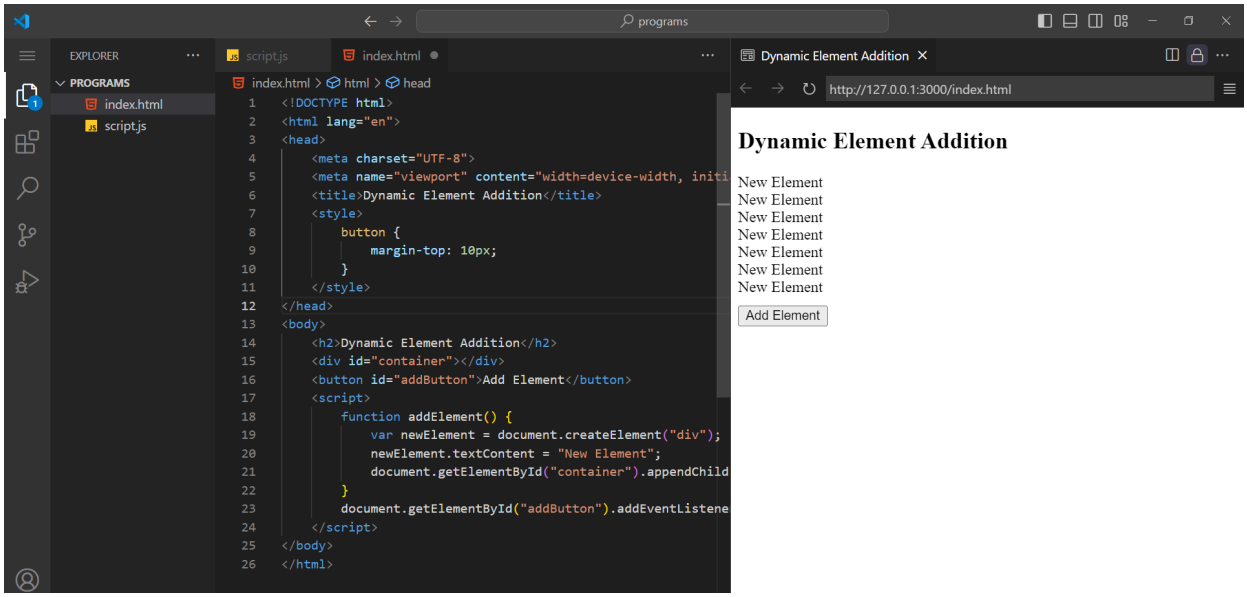
- **What is negative Infinity?**
  - The negative infinity in JavaScript is a constant value that is used to represent a value that is the lowest available. This means that no other number is less than this value. It can be generated using a self-made function or by an arithmetic operation.

- **Which company developed JavaScript?**
  - JavaScript was created and developed at Netscape Communications by Brendan Eich in 1995. Netscape and Eich designed JavaScript as a scripting language for use with the company's flagship web browser, Netscape Navigator.

- **What are undeclared and undefined variables?**
  - Declared variables are created before any code is executed. Declared variables are a non-configurable property of their execution context (function or global).
  - Undeclared variables do not exist until the code assigned to them is executed. Undeclared variables are configurable (e.g. can be deleted).

- **Write the code for adding new elements dynamically?**





- **What is the difference between ViewState and SessionState?**

View state

- ViewState is used to preserve the state of page and control values across HTTP requests.
- It stores data of a single page in a hidden field on the page itself.
- ViewState is useful for maintaining state between postbacks of a single page.
- It works at the control level, meaning each control can maintain its own ViewState.
- ViewState is preserved even if the page is refreshed.

- ViewState is lightweight compared to SessionState.

Session state

- SessionState is used to store user-specific information across multiple HTTP requests.
- It stores data across all pages in a user session, which is typically maintained either in-memory (InProc), in a separate process (StateServer), or in a database (SQLServer).
- SessionState is useful for maintaining user-specific information like login details, user preferences, etc.
- It works at the session level, meaning the data stored in SessionState is available across multiple pages during a user's session.
- SessionState is terminated when the session expires (usually due to inactivity) or when the user closes the browser.
- SessionState requires server resources to store session data, so it can impact scalability if not managed properly.

• What is === operator?

- The === operator in JavaScript is called the strict equality operator. It checks if two values are equal in both value and type without performing type coercion. If the values being compared are of different types, the result is false.
-

• How can the style/class of an element be changed?

- The style of an element can be changed using the style property or by adding/removing classes.
- Example:

```
// Changing style directly
document.getElementById('myElement').style.color = 'red';
// Adding a class
document.getElementById('myElement').classList.add('highlight');
// Removing a class
document.getElementById('myElement').classList.remove('highlight');
```

• How to read and write a file using JavaScript?

• What are all the looping structures in JavaScript?

- for loop
- while loop
- do...while loop
- for...in loop
- for...of loop (introduced in ES6)

• How can you convert the string of any base to an integer in JavaScript?

- You can use the parseInt() function to convert a string representation of a number in any base to an integer.
- Example:
  parseInt('1010', 2);

• What is the function of the delete operator?

- The delete operator is used to remove a property from an object.
- Example:
  const obj = {a: 1, b: 2};
  delete obj.a;

• What are all the types of Pop up boxes available in JavaScript?

- alert(): Displays an alert box with a message and an OK button.
- confirm(): Displays a dialog box with a message and OK and Cancel buttons.
- prompt(): Displays a dialog box that prompts the user for input.

• What is the use of Void (0)?

- void(0) is used to return undefined. It is often used in the href attribute of an anchor tag to prevent the page from navigating anywhere when the link is clicked.
- Example:
  Click me

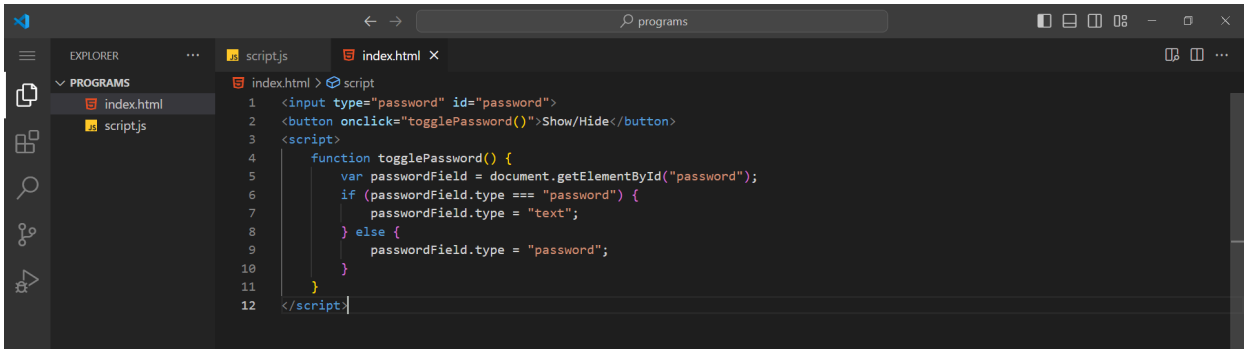
• How can a page be forced to load another page in JavaScript?

- You can use window.location to redirect the browser to another page.
- Example:
  window.location.href = 'https://www.example.com';

• What are the disadvantages of using innerHTML in JavaScript?

- Security risk : Inserting unsanitized user input via innerHTML can lead to cross-site scripting attacks.
- Performance : Replacing the entire HTML content of an element using innerHTML is less efficient than using DOM manipulation methods like createElement and appendChild.
- Event listeners : If elements with event listeners are replaced using innerHTML, those listeners might get detached and need to be reattached.

• Create password field with show hide functionalities


programs
EXPLORER
PROGRAMS
index.html
script.js
script.js
index.html
index.html > script
<input type="password" id="password">
<button onclick="togglePassword()">Show/Hide</button>
<script>
function togglePassword() {
var passwordField = document.getElementById("password");
if (passwordField.type === "password") {
passwordField.type = "text";
} else {
passwordField.type = "password";
}
}
</script>


index.html
http://127.0.0.1:3000/index.html
Show/Hide


index.html
http://127.0.0.1:3000/index.html
sdvbgnnnh
Show/Hide