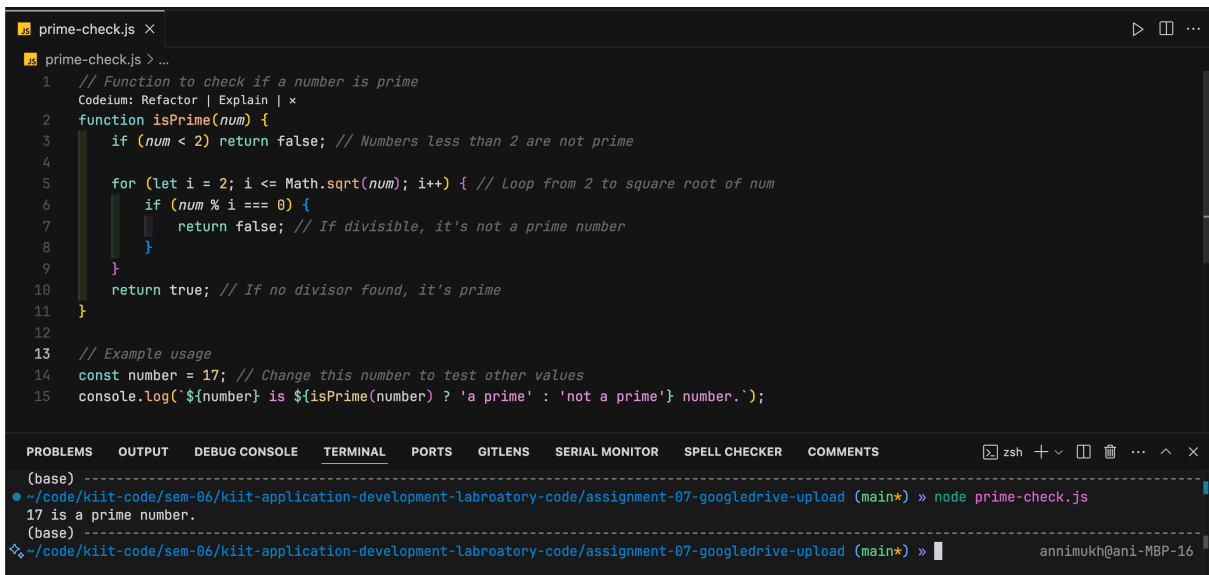


Application Development Lab Assignment

Submitted by **Aniruddha Mukherjee [2205533]** CSE-16

Q1:

Design A JavaScript to display whether given number is prime or not.



```
1 // Function to check if a number is prime
2 function isPrime(num) {
3     if (num < 2) return false; // Numbers less than 2 are not prime
4
5     for (let i = 2; i <= Math.sqrt(num); i++) { // Loop from 2 to square root of num
6         if (num % i === 0) {
7             return false; // If divisible, it's not a prime number
8         }
9     }
10    return true; // If no divisor found, it's prime
11 }
12
13 // Example usage
14 const number = 17; // Change this number to test other values
15 console.log(`${number} is ${isPrime(number) ? 'a prime' : 'not a prime'} number.`);
```

Terminal output:

```
(base) ~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-googledrive-upload (main*) » node prime-check.js
17 is a prime number.
(base) ~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-googledrive-upload (main*) »
```

Q2:

Explain about Function definition, Function calling, Function parameter, return type with a suitable program in JavaScript.

```
2-function-demo.js X
2-function-demo.js > ...
1 // Function definition
  Codeium: Refactor | Explain | x
2 function addNumbers(a, b) {
3   return a + b; // Returns the sum of a and b
4 }
5
6 // Function call
7 const sum = addNumbers(5, 10);
8 console.log(`Sum: ${sum}`); // Output the sum

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SERIAL MONITOR SPELL CHECKER COMMENTS
(base)
~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-google-drive-upload (main*) » node 2-function-demo.js
Sum: 15
(base)
~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-google-drive-upload (main*) » annimukh@ani-MBP-16
```

Q3:

Explain about Cascading Style Sheets with a program

```
3-css-demo.html X
3-css-demo.html > html < head < style
2 <html lang="en">
3 <head>
7 <style>
13 .container {
14   display: inline-block;
20 }
21 h1 {
22   color: #333;
23 }
24 </style>
25 </head>
26 <body>
27 <div class="container">
28   <h1>Welcome to CSS Styling</h1>
29   <p>This is an example of inline CSS styling.</p>
30 </div>
31 </body>
32 </html>
33
34

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ...
(base)
~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-google-drive-upload (main*) » open 3-css-demo.html
(base)
~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-google-drive-upload (main*) » annimukh@ani-MBP-16
```

Q4:

Explain various operators and data types available in java script with examples

```
4-operators-data-types.js X
4-operators-data-types.js > ...
1 // Arithmetic Operators
2 let a = 10, b = 5;
3 console.log('Addition: ${a + b}');
4 console.log('Subtraction: ${a - b}');
5 console.log('Multiplication: ${a * b}');
6 console.log('Division: ${a / b}');
7
8 // Comparison Operators
9 console.log('Equal: ${a == b}');
10 console.log('Not Equal: ${a != b}');
11 console.log('Greater than: ${a > b}');
12
13 // Logical Operators
14 console.log('AND: ${true && false}');
15 console.log('OR: ${true || false}');
16 console.log('NOT: ${!true}');
17
18 // Data Types
19 let str = "Hello"; // String
20 var boolean = true; // Boolean
21 let num = 42; // Number
22 let obj = { name: "John" }; // Object
23 console.log('String: ${str}, Boolean: ${boolean}, Number: ${num}, Object: ${JSON.stringify(obj)}');
24

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SERIAL MONITOR SPELL CHECKER COMMENTS
(base) -----
• ~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-google-drive-upload (main*) » node 4-operators-data-types.js
Addition: 15
Subtraction: 5
Multiplication: 50
Division: 2
Equal: false
Not Equal: true
Greater than: true
AND: false
OR: true
NOT: false
String: Hello, Boolean: true, Number: 42, Object: {"name":"John"}
(base) -----
~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-google-drive-upload (main*) » | annimukh@ani-MBP-16
```

Q5:

Build a JavaScript program to convert temperature from Celsius to Fahrenheit and vice versa

```

5-temperature-conversion.js x
5-temperature-conversion.js > ...
1 // Function to convert Celsius to Fahrenheit
  Codeium: Refactor | Explain | x
2 function celsiusToFahrenheit(celsius) {
3   return (celsius * 9/5) + 32;
4 }
5
6 // Function to convert Fahrenheit to Celsius
  Codeium: Refactor | Explain | x
7 function fahrenheitToCelsius(fahrenheit) {
8   return (fahrenheit - 32) * 5/9;
9 }
10
11 // Example usage
12 let celsius = 25;
13 console.log(`${celsius}°C = ${celsiusToFahrenheit(celsius)}°F`);
14
15 let fahrenheit = 77;
16 console.log(`${fahrenheit}°F = ${fahrenheitToCelsius(fahrenheit)}°C`);
17

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SERIAL MONITOR ...
(base) -----
~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-google-drive-upload (main*) » node 5-temperature-conversion.js
25°C = 77°F
77°F = 25°C
(base) -----
~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-google-drive-upload (main*) »

```

Q6:

Define Form tag. Design a Registration page by using all Form controls

```

6-registration-form.html x
6-registration-form.html > html > body > form > br
2 <html>
6 <body>
8   <form action="#" method="post">
9     <label for="name">Name:</label>
10    <input type="text" id="name" name="name" required><br><br>
11
12    <label for="email">Email:</label>
13    <input type="email" id="email" name="email" required><br><br>
14
15    <label for="password">Password:</label>
16    <input type="password" id="password" name="password" required><br><br>
17
18    <label for="gender">Gender:</label>
19    <select id="gender" name="gender">
20      <option value="male">Male</option>
21      <option value="female">Female</option>
22    </select><br><br>
23
24    <input type="submit" value="Register">
25  </form>
26 </body>
27 </html>
28

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ...
(base) -----
~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-google-drive-upload (main*) » open 6-registration-form.html
~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-google-drive-upload (main*) »

```

Registration Form

Name:

Email:

Password:

Gender: Male

Q7:

Define Table tag and their attributes with an example.

The screenshot shows a code editor with a file named `7-table-example.html`. The code defines an HTML document with a table. The table has a border attribute set to "1". It contains two rows of data. The first row has the values "John", "25", and "New York". The second row has the values "Jane", "30", and "Los Angeles".

```
2 <html>
6 <body>
8   <table border="1">
9     <tr>
13      <td>John</td>
14    </tr>
15    <tr>
16      <td>John</td>
17      <td>25</td>
18      <td>New York</td>
19    </tr>
20    <tr>
21      <td>Jane</td>
22      <td>30</td>
23      <td>Los Angeles</td>
24    </tr>
25  </table>
26 </body>
27 </html>
```

Below the code editor, a terminal window shows the command `open 7-table-example.html` being executed. To the right of the code editor, a preview of the rendered table is shown, titled "Sample Table".

Name	Age	City
John	25	New York
Jane	30	Los Angeles

Q8:

Write a JavaScript program to print all indices of an array where perfect numbers are present.

```
8-perfect-numbers.js > ...  
2 function isPerfect(num) {  
3   let sum = 0;  
4   for (let i = 1; i < num; i++) {  
5     if (num % i === 0) {  
6       sum += i;  
7     }  
8   }  
9   return sum === num;  
10 }  
11  
12 // Function to find indices of perfect numbers in an array  
Codeium: Refactor | Explain | x  
13 function findPerfectIndices(arr) {  
14   return arr.map((num, index) => isPerfect(num) ? index : -1).filter(index => index !== -1);  
15 }  
16  
17 // Example usage  
18 let numbers = [6, 28, 12, 496, 8128, 7];  
19 console.log(`Indices of perfect numbers: ${findPerfectIndices(numbers)}`);  
20  
  
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SERIAL MONITOR ...  
(base) -----  
• ~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-googledrive-upload (main*) » node 8-perfe  
ct-numbers.js  
Indices of perfect numbers: 0,1,3,4  
(base) -----  
❖ ~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-googledrive-upload (main*) »
```

Q9:

Write a JavaScript program to validate e-mail id, phone number and password

```
9-validation.js x
9-validation.js > ...
1 // Function to validate email
  Codeium: Refactor | Explain | x
2 function validateEmail(email) {
3     const regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
4     return regex.test(email);
5 }
6
7 // Function to validate phone number (10-digit format)
  Codeium: Refactor | Explain | x
8 function validatePhone(phone) {
9     const regex = /^\d{10}$/;
10    return regex.test(phone);
11 }
12
13 // Function to validate password (at least 8 characters, one number, one special character)
  Codeium: Refactor | Explain | x
14 function validatePassword(password) {
15     const regex = /^(?=.*[A-Za-z])(?=.*\d)(?=.*[@$!%*?&])[A-Za-z\d@$!%*?&]{8,}$/;
16     return regex.test(password);
17 }
18
19 // Example usage
20 const email = "test@example.com";
21 const phone = "1234567890";
22 const password = "Test@1234";
23
24 console.log(`Email valid: ${validateEmail(email)}`);
25 console.log(`Phone valid: ${validatePhone(phone)}`);
26 console.log(`Password valid: ${validatePassword(password)}`);
27
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SERIAL MONITOR ...
zsh + - [ ] [ ] ... ^ x
(base) -----
• ~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-google-drive-upload (main*) » node 9-validation.js
Email valid: true
Phone valid: true
Password valid: true
(base) -----
❖ ~/code/kiit-code/sem-06/kiit-application-development-labroatory-code/assignment-07-google-drive-upload (main*) »
```