Week-4

Nischal k
497CS23033

1.Create a form like registration form, after submit hide create form and enable the display section.

```
//Input
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Registration Form</title>
   <style>
.hidden {
          display: none;
      }
   </style>
</head>
<body>
   <div id="form">
      <form id="registration">
         <input type="text" id="name" placeholder="Name" required>
         <input type="number" id="name" placeholder="Age" required>
         <input type="email" id="email" placeholder="Email" required>
         <button type="submit">Submit</button>
      </form>
   </div>
   <div id="thankyou" class="hidden">
      <h3>Thank you for your submission!</h3>
   </div>
   <script>
      document.getElementById('registration').addEventListener('submit', function(event)
{           event.preventDefault();
document.getElementById('form').classList.add('hidden');
document.getElementById('thankyou').classList.remove('hidden');
      });
   </script>
</body>
</html>
//output
```

Thank you for your submission!

2.Write a program to differentiate Types of CSS

CSS is used to style and layout web pages, controlling the appearance of HTML elements. It allows developers to create visually appealing designs and ensure a consistent look across a website.

Types of CSS

CSS can be implemented in three different ways:

- Inline CSS
- Internal or Embedded CSS
- External CSS

* Inline CSS

Inline CSS involves applying styles directly to individual HTML elements using the style attribute. This method allows for specific styling of elements within the HTML document, overriding any external or internal styles.

```
//input
<html>
<head>
</head>
<body>
<p style="color:#009900;  font-size:50px;    font-style:italic;  text-align:center;">Inline CSS</p>
</body>
</html>
```

//output

Inline CSS

* Internal CSS

```
//input
<!DOCTYPE html>
<html>
<head>
<style>
.cent {
color:red;  text-
align:center;
}
p.cent {
 font-size:300%;
}
p.large {
```

```
 font-size:300%;
}
</style>
</head>
<body>
<h1 class="cent">hello world </h1>
<p class="cent"> welcome to SDM polytechnic </p>
<p class="large"> welcome to cs department </p>
</body>
</html>
```

//output

hello world

welcome to SDM polytechnic

welcome to cs department

* External CSS

//Input

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="mystyle.css"/>
</head>
<body>
<h1 class="header">hello world </h1>
<p class ="para"> welcome to SDM polytechnic </p>
</body>
</html>
```

mystyle.css //Input

```
.header {
color:red;  text-
align:center;
}
.para {  font-
size:12pt;
color:red; }
```

//output

hello world

welcome to SDM polytechnic

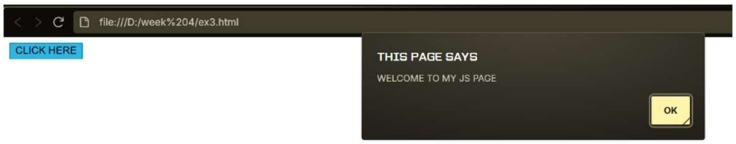
3.Write a JS program to demonstrate functions.

- Functions are fundamental building blocks in all programming.
- Functions enable better code organization, modularity, and efficiency.
- Functions are reusable blocks of code designed to perform a particular task.

- Functions execute when they are "called" or "invoked".
- A function is defined with the function keyword, followed by the function name, followed by parentheses ( ), followed by brackets { }.
- The name follows the naming rules for variables (letters, digits, ...). The code to be executed is listed inside curly brackets: { code } Functions can optionally return a value back to the "caller".

```
 //input
<html>
<head>
<title> js demo</title> <script
type="text/javascript">
function greet()
{
 alert("WELCOME TO MY JS PAGE");
}
</script>
</head>
<body>
<button onclick="greet();">CLICK HERE</BUTTON>
</body>
</html>
```

//output



4.Create a JSON object in javascript and print the contents of the object.

- SON stands for JavaScript Object Notation
- JSON is a text format for storing and transporting data
- JSON is "self-describing" and easy to understand This example is a JSON string: '{"name":"John", "age":30, "car":null}' It defines an object with 3 properties:
- name
- age car

Each property has a value.

If you parse the JSON string with a JavaScript program, you can access the data as an object: Since the format is text only, JSON data can easily be sent between computers, and used by any programming language.

JavaScript has a built in function for converting JSON strings into JavaScript objects: JSON.parse()

JavaScript also has a built in function for converting an object into a JSON string: JSON.stringify()

Ex:1 JSON.parse()

//input

```
<html>
   <body>
      <h1>json converting to object</h1>
      <p id="demo"> hello</p>
      <script>
         const txt='{"name":"Nischal k", "age":17, "place":"Mysore"}';
const myobj=JSON.parse(txt);
         document.getElementById("demo").innerHTML=myobj.name;
      </script>
   </body>
</html>
```

//output

json converting to object

Nischal k

Ex2: JSON.stringify()

```
//input
<html>
   <body>
      <h1>json converting to string using stringify</h1>
      <p id="demo"> Hello</p>
      <script>
         const person={"name":"Nischal k", "age":17, "place":"Mysore"};

         const txt=JSON.stringify(person);
document.getElementById("demo").innerHTML=txt;
      </script>
   </body>
</html>
```

//output

json converting to string using stringify

{"name":"Nischal k","age":18,"place":"Mysore"}

4. Using ES6, write a program to explain arrow function.

- Arrow functions were introduced in ES6.
- Arrow functions allow a shorter syntax for function expressions.
- You don't need the function keyword, the return keyword, and the curly brackets:

//input

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Functions</h1>
<h2>The Arrow Function</h2>
<p>This example shows the syntax of an Arrow Function, and how to use it.</p>
<p id="demo"></p>
<script> let hello = ()
=> {   return "Hello
World!";
} document.getElementById("demo").innerHTML =
hello();
</script>
</body>
</html>
//output
```

JavaScript Functions

The Arrow Function

This example shows the syntax of an Arrow Function, and how to use it.

Hello World!