Assignment 5A

Aim: Write a JavaScript Program to study DOM Manipulation and CSS Manipulations.

Theory:

1. DOM Manipulation:

   The Document Object Model (DOM) is a programming interface for web documents. It represents the structure of a web page as a tree of objects, where each element, attribute, and piece of text is a node in the tree. DOM manipulation involves using JavaScript to interact with and modify the content, structure, and style of a web page.
   In JavaScript, you can use various methods and properties to manipulate the DOM.

   Some key concepts include:
   - Selecting Elements: You can select elements using methods like `document.querySelector()` or `document.getElementById()`.
   - Creating Elements: You can create new DOM elements using `document.createElement()` and then append them to the DOM tree with methods like `appendChild()`.
   - Modifying Content: You can change the text or HTML content of an element using properties like `textContent` or `innerHTML`.
   - Modifying Attributes: You can change attributes of elements using methods like `setAttribute()`.
   - Adding Event Listeners: You can attach event listeners to elements to respond to user interactions like clicks, mouseover etc.

2. CSS Manipulation

   Cascading Style Sheets (CSS) is used to define the presentation and layout of web documents. CSS manipulation involves using JavaScript to change the styles of elements on a web page dynamically. In JavaScript, you can manipulate CSS styles using the `style` property of DOM elements.

   Some key points include:
   - Changing Styles: You can modify CSS properties like
     `backgroundColor`, `color`, `width`, etc using the `style` property.
   - Adding/Removing Classes: You can add or remove classes from elements using methods like `classList.add()` and `classList.remove()`.
   - Pseudo-Classes: You can target pseudo-classes like `:hover`, `:active`, etc., to apply styles based on user interactions.
   - Transitions and Animations: You can create smooth transitions and animations by changing CSS properties with JavaScript over time.

Combining DOM manipulation and CSS manipulation allows you to create dynamic and interactive web pages. These techniques are commonly used to build web applications, enhance user experience, and create visually appealing interfaces.

It's important to note that these manipulations are often performed within an HTML context, as the DOM and CSS styles are integral parts of web development. The examples provided earlier showcase how these concepts work together to create interactive and visually appealing elements on a web page.

Code snapshot:

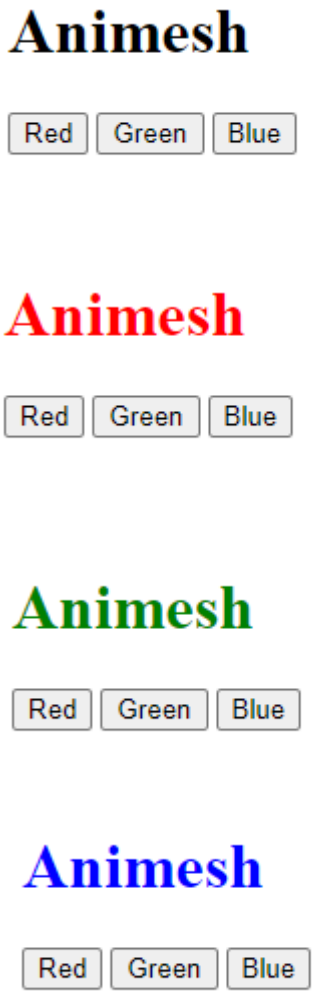
```
<!DOCTYPE html>
<html>
<head>

</head>
<body>
  <h1 id="heading">Animesh</h1>
  <button id="redButton">Red</button>
  <button id="greenButton">Green</button>
  <button id="blueButton">Blue</button>

  <script>
    const heading = document.getElementById('heading');
    const redButton = document.getElementById('redButton');
    const greenButton = document.getElementById('greenButton');
    const blueButton = document.getElementById('blueButton');

    redButton.addEventListener('click', () => changeColor('red'));
    greenButton.addEventListener('click', () => changeColor('green'));
    blueButton.addEventListener('click', () => changeColor('blue'));

    function changeColor(color) {
      heading.style.color = color;
    }
  </script>
</body>
</html>
```



Conclusion:

In conclusion, DOM manipulation and CSS manipulation are two fundamental techniques in JavaScript that empower developers to dynamically interact with and enhance web pages. DOM manipulation involves using JavaScript to navigate, create, modify, and remove elements in the Document Object Model, enabling dynamic content updates and responsive user interactions. CSS manipulation, on the other hand, allows for the modification of element styles, enabling the creation of visually engaging and interactive designs. By combining these techniques, developers can create immersive web experiences, modify content on-the-fly, and tailor user interfaces to deliver seamless and engaging interactions on the modern web.