

| Title: Implementation of Advanced JavaScript Concept |
| --- |

**AIM:** To Implement the Concept of Advanced JavaScript

**Problem Definition:**

-Demonstrate the Concept of Advanced JavaScript With the help of Example.

\*(Students have to perform the task assigned within group and demonstrate the same).

**Resources used:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected OUTCOME of Experiment:**

**CO 1:**.Build full stack applications in JavaScript using the MERN technologies.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. Shelly Powers Learning Node O’ Reilly 2 nd Edition, 2016.

**Pre Lab/ Prior Concepts:**

**Methodology:**

Task 1: Array Manipulations with `styles` Array

1. Create an array `styles` with items "Jazz" and "Blues":

- Initialize an array with the given items.

2. Append "Rock-n-Roll" to the end:

- Use the `push` method to add "Rock-n-Roll" to the end of the array.

3. Replace the value in the middle with "Classics":

- Find the middle index of the array using `Math.floor` and array length. Replace the element at that index.

4. Strip off the first value of the array and show it:

- Use the `shift` method to remove and return the first element of the array.

5. Prepend "Rap" and "Reggae" to the array:

- Use the `unshift` method to add elements to the beginning of the array.

6. Create a new array with numbers from 1 to 10. Use `splice` to remove the numbers at positions 4 to 6 and replace them with "a", "b", "c". Display the modified array:

- Initialize an array with numbers 1 to 10. Use the `splice` method to replace specific elements.

7. Combine the original array with another array `["m", "n", "o"]` using `concat` and display the result:

- Use the `concat` method to combine two arrays.

Task 2: User Input for Array Manipulations

1. Get user inputs for initial styles and other operations:

- Define utility functions `getUserInputArray` and `getUserInputNumber` to gather and process user inputs.

2. Execute the array manipulations with user inputs:

- Use the functions above to receive inputs for array manipulations, such as adding new styles, replacing elements, and combining arrays.

Task 3: Unique Array Members

1. Define a function `unique` to return an array of unique items:

- Use a `Set` to store unique values and convert it back to an array.

2. Example usage:

- Demonstrate the use of the function with an example array.

Task 4: Filter Anagrams

1. Define a function `areAnagrams` to check if two words are anagrams:

- Compare sorted versions of the words.

2. Define a function `filterAnagrams` to filter anagrams from a list of words:

- Use the `filter` method with `areAnagrams` to return words that are anagrams of the target word.

3. Example usage:

- Demonstrate the function with an example list of words and a target word.

Task 5: Iterable Keys from a Map

1. Create a `Map` and retrieve its keys:

- Use `Array.from` or spread operator to convert `map.keys()` iterator to an array.

2. Apply array-specific methods to the array of keys:

- Demonstrate adding elements using `push`.

Task 6: Fetch Users from GitHub

1. Define an async function `getUsers` to fetch GitHub user data:

- Use `fetch` to make API requests and `Promise.all` to wait for all promises to resolve.

2. Example usage:

- Demonstrate fetching user data for a list of usernames.

**Implementation Details:**

Students have to write stepwise details of implementation.

**Code:**

// 1. Create an array styles with items “Jazz” and “Blues”.

let styles = ["Jazz", "Blues"];

// 2. Append “Rock-n-Roll” to the end.

styles.push("Rock-n-Roll");

// 3. Replace the value in the middle with “Classics”.

let middleIndex = Math.floor(styles.length / 2);

styles[middleIndex] = "Classics";

// 4. Strip off the first value of the array and show it.

let firstValue = styles.shift();

console.log("Removed first value:", firstValue);

// 5. Prepend Rap and Reggae to the array.

styles.unshift("Rap", "Reggae");

console.log("Updated styles array:", styles);

// 6. Create a new array with numbers from 1 to 10. Use splice to remove the numbers at positions 4 to 6 and replace them with "a", "b", "c". Display the modified array.

let numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

numbers.splice(4, 3, "a", "b", "c");

console.log("Modified numbers array:", numbers);

// 7. Combine the original array with another array ["m", "n", "o"] using concat and display the result.

let combinedArray = numbers.concat(["m", "n", "o"]);

console.log("Combined array:", combinedArray);

function getUserInputArray(promptMessage)

{

    let input = prompt(promptMessage);

    return input.split(',').map(item => item.trim());

}

function getUserInputNumber(promptMessage)

{

    let input = prompt(promptMessage);

    return Number(input);

}

let styles = getUserInputArray("Enter initial styles separated by commas (e.g., Jazz, Blues):");

let newStyle = prompt("Enter a new style to append to the array:");

styles.push(newStyle);

let middleIndex = Math.floor(styles.length / 2);

let newMiddleValue = prompt(`Enter a new style to replace the middle value (${styles[middleIndex]}):`);

styles[middleIndex] = newMiddleValue;

let firstValue = styles.shift();

console.log("Removed first value:", firstValue);

let prependStyles = getUserInputArray("Enter styles to prepend to the array separated by commas (e.g., Rap, Reggae):");

styles.unshift(...prependStyles);

console.log("Updated styles array:", styles);

let numbers = Array.from({ length: 10 }, (\_, i) => i + 1);

let replacements = getUserInputArray("Enter three values to replace positions 4 to 6 in the array separated by commas (e.g., a, b, c):");

numbers.splice(4, 3, ...replacements);

console.log("Modified numbers array:", numbers);

let additionalArray = getUserInputArray("Enter additional values to combine with the numbers array separated by commas (e.g., m, n, o):");

let combinedArray = numbers.concat(additionalArray);

console.log("Combined array:", combinedArray);

**Code:**

function unique(arr) {

    return [...new Set(arr)];

}

// Example usage:

const array = [1, 2, 2, 3, 4, 4, 5];

console.log(unique(array)); // Output: [1, 2, 3, 4, 5]

function areAnagrams(word1, word2) {

    const sortedWord1 = word1.split('').sort().join('');

    const sortedWord2 = word2.split('').sort().join('');

    return sortedWord1 === sortedWord2;

}

function filterAnagrams(words, target) {

    return words.filter(word => areAnagrams(word, target));

}

// Example usage:

const words = ['listen', 'silent', 'enlist', 'inlets', 'hello'];

const target = 'listen';

console.log(filterAnagrams(words, target)); // Output: ['listen', 'silent', 'enlist', 'inlets']

const myMap = new Map([

    [1, 'one'],

    [2, 'two'],

    [3, 'three']

]);

const keysArray = Array.from(myMap.keys()); // or [...myMap.keys()]

// Apply array-specific methods

keysArray.push(4);

console.log(keysArray); // Output: [1, 2, 3, 4]

async function getUsers(names) {

    const requests = names.map(name => fetch(`https://api.github.com/users/${name}`));

    const responses = await Promise.all(requests);

    const users = await Promise.all(responses.map(response => response.json()));

    return users;

}

// Example usage:

const usernames = ['octocat', 'torvalds'];

getUsers(usernames)

    .then(users => console.log(users))

    .catch(error => console.error('Error fetching users:', error));

**Steps for execution:**

Here are the short execution steps for the provided JavaScript tasks:

1. Array Manipulations (`styles` Array):

- Initialize: Start with an array `styles` containing "Jazz" and "Blues".

- Append: Add "Rock-n-Roll" to the end of the array.

- Replace Middle: Calculate the middle index and replace the element with "Classics".

- Strip First Value: Remove and display the first element using `shift`.

- Prepend: Add "Rap" and "Reggae" to the beginning using `unshift`.

- Splice Array: Create an array of numbers from 1 to 10, replace positions 4 to 6 with "a", "b", "c" using `splice`.

- Concat Arrays: Combine the modified numbers array with `["m", "n", "o"]` using `concat`.

2. User Input for Array Manipulations:

- Gather Inputs: Use prompt dialogs to get user inputs for initial array items, new styles, middle replacement, and additional arrays.

- Perform Operations: Execute the array manipulations using the provided inputs.

3. Unique Array Members:

- Define `unique` Function: Implement a function that returns an array of unique elements using a `Set`.

4. Filter Anagrams:

- Define `areAnagrams` Function: Check if two words are anagrams by comparing sorted character arrays.

- Filter Anagrams: Use `filter` method with `areAnagrams` to find anagrams in a list.

5. Iterable Keys from a Map:

- Create Map and Extract Keys: Initialize a `Map`, extract keys using `Array.from` or spread operator, and store them in an array.

- Modify Keys Array: Apply array methods like `push` on the keys array.

6. Fetch Users from GitHub:

- Define `getUsers` Function: Implement an async function that fetches user data from GitHub using their usernames.

- Execute and Display: Fetch user data, handle responses, and display the results.

**Conclusion:**

**We learned to implement advance functions in JavaScript.**