Experiment No._____ *Date___/___/2020*

TITLE OF EXPERIMENT: - A Program to create an array and perform following operations

- **To remove specific element from the array.**
- **Check if an array contains a specified value.**
- **To empty an array**

DIVISION:_____ **BRANCH:** _____

BATCH:_____ **ROLL NO.:** _____

PERFORMED ON DATE: _____

SIGNATURE OF TEACHING STAFF:

EXPERIMENT NO. 6

Aim: Write a JavaScript program that will create an array and perform following operations
• To remove specific element from the array.
• Check if an array contains a specified value.
• To empty an array

Prerequisites:

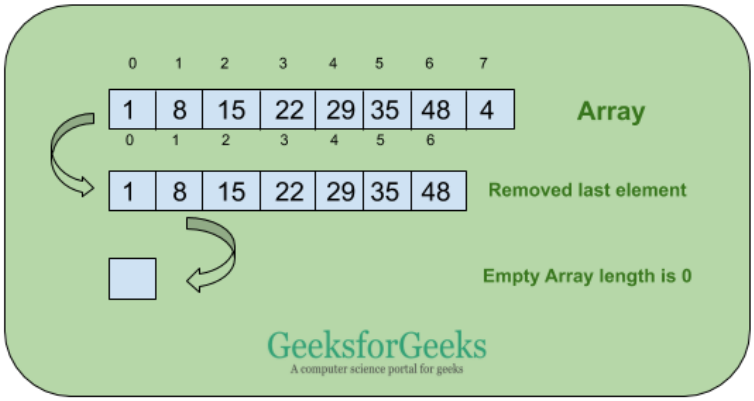
- For this **Javascript Lab**, it is assumed that you have a prior knowledge of HTML coding. It would help if you had some prior exposure to object-oriented programming concepts and a general idea on creating online applications.
- To understand this experiment, you should have the knowledge of the basic JavaScript, JavaScript Array includes(), JavaScript Array indexOf(), JavaScript Arrays

Editor:

1.	NotePad
2.	Visual studio code

Theory:

A) To remove specific element from the array.



JavaScript array is a single variable that is used to store the elements or a group of values. You can add or remove elements from array in any position. In this experiment, we will discuss different ways to remove elements from array. There are many methods that is used to remove elements from JavaScript array which are discussed below:

- **pop() function:** This method is use to remove elements from the end of an array.
- **shift() function:** This method is use to remove elements from the start of an array.
- **splice() function:** This method is use to remove elements from the specific index of an array.
- **filter() function:** This method is use to remove elements in programmatically way.

Note: There are some other methods that are created by JavaScript inbuilt methods. Below examples illustrate the methods to remove elements from a JavaScript array:

1) Remove Array elements by using pop() method: This method is used to remove the last element of the array and returns the removed element. This function decreases the length of the array by 1.

Example 1:

```
<script>

// JavaScript code to illustrate pop() function

// to remove array elements



function func() {

    var arr = ["shift", "splice", "filter", "pop"];
```

```
    // Popping the last element from the array
    var popped = arr.pop();
    document.write("Removed element: " + popped + "<br>");
    document.write("Remaining elements: " + arr);
}
func();
</script>
```

Output:

```
Removed element: pop
Remaining elements: shift, splice, filter
```

Example 2:

```
<script>
// Declare and initialize an array
var array = ["pop", "splice", "filter", "shift"]

document.write("Original array: " + array + "<br>")

// Loop run while array length not zero
```

```
while (array.length) {



    // Remove elements from array

    array.pop();

}

document.write("Array Length: " + array.length )



</script>
```

Output:

```
Original array: pop, splice, filter, shift
Array Length: 0
```

2) Remove Array elements by using shift() method: This method is used to remove the first element of the array and reducing the size of original array by 1.

Example:

```
<script>

// JavaScript code to illustrate shift() method

// to remove elements from array

function func() {

    var arr = ["shift", "splice", "filter", "pop"];



    // Removing the first element from array
```

```
    var shifted = arr.shift();

    document.write("Removed element: " + shifted + "<br>");

    document.write("Remaining elements: " + arr);

}

func();

</script>
```

Output:

```
Removed element: shift
Remaining elements: splice, filter, pop
```

3) Remove Array elements by using splice() method: This method is used to modify the contents of an array by removing the existing elements and/or by adding new elements. To remove elements by splice() method you can specify the elements in different ways.

Example 1: Use the indexing of splice method to remove elements from a JavaScript array.

```
<script>

// JavaScript code to illustrate splice() function



function func() {

    var arr = ["shift", "splice", "filter", "pop"];



    // Removing the specified element from the array
```

```
    var spliced = arr.splice(1, 1);

    document.write("Removed element: " + spliced + "<br>");

    document.write("Remaining elements: " + arr);

}

func();

</script>
```

Output:

```
Removed element: splice
Remaining elements: shift, filter, pop
```

4) Remove Array elements by using filter() method: This method is used to create a new array from a given array consisting of only those elements from the given array which satisfy a condition set by the argument function. To remove elements by filter() method you can specify the elements in different ways.

Example: Use the value of filter method to remove elements from a JavaScript array.

```
<script>



// JavaScript to illustrate filter() method

function isPositive( value ) {

    return value > 0;

}
```

```
function func() {

    var filtered = [101, 98, 12, -1, 848].filter( isPositive );

    document.write("Positive elements in array: " + filtered);

}

func();

</script>
```

Output:

```
Positive elements in array: 101, 98, 12, 848
```

5) Remove Array elements by using Remove Method: Creating a remove method using filter method to remove elements from a JavaScript array. This methods works in reverse order.

Example:

```
<script>



// Declare and initialize an array

var array = ["lowdash", "remove", "delete", "reset"]



// Using filter method to create a remove method

function arrayRemove(arr, value) {



   return arr.filter(function(geeks){
```

```
    return geeks != value;
  });


}


var result = arrayRemove(array, "delete");
document.write("Remaining elements: " + result)
</script>
```

Output:

```
Remaining elements: lowdash, remove, reset
```

6) Remove Array elements by Delete Operator: Use the delete operator to remove elements from a JavaScript array.

Example:

```
<script>


// Declare and initialize an array
var array = ["lowdash", "remove", "delete", "reset"]


// Delete element at index 2
var deleted = delete array[2];
```

```
document.write("Removed: " + deleted + "<br>");

document.write("Remaining elements: " + array);

</script>
```

Output:

```
Removed: true
Remaining elements: lowdash, remove,,reset
```

7) Remove Array elements by Clear and Reset Operator: Use clear and reset operator to remove elements from a JavaScript array.

Example 1:

```
<script>



// Declare and initialize an array

var array = ["lowdash", "remove", "delete", "reset"]



// Sorting array in another array

var arraygeeks = array



// Delete each element of array

array = []

document.write("Empty array: " + array + "<br>")
```

```
document.write("Original array: " + arraygeeks)



</script>
```

Output:

```
Empty array:
Original array: lowdash, remove, delete, reset
```

8) Remove Array elements using a simple for() loop and a new array: Here a simple for() will be run over the array and the except for the removed element, remaining elements will be pushed into the new array which will be declared inside the function or a method.

Example:

```
let removeElement = (array, n) => {
  let newArray = [];



  for (let i = 0; i < array.length; i++) {
    if (array[i] !== n) {
      newArray.push(array[i]);
    }
  }
  return newArray;
};



let passed_in_array = [1, 2, 3, 4, 5];
```

```
let element_to_be_removed = 2;

let result = removeElement(passed_in_array, element_to_be_removed);



console.log("Remaining elements: " + result);



// This code is contributed by Aman Singla...
```

Output:

```
Remaining elements: 1,3,4,5
```

B) Check if an array contains a specified value.

Check Array Using includes()

```
// program to check if an array contains a specified value

const array = ['you', 'will', 'learn', 'javascript'];

const hasValue = array.includes('javascript');

// check the condition
if(hasValue) {
   console.log('Array contains a value.');
} else {
   console.log('Array does not contain a value.');
}
```

Output

```
Array contains a value.
```

In the above program, the `includes()` method is used to check if an array contains a specified value.

- The `includes()` method returns `true` if the value exists in the array.

- The if...else statement is used to display the result as per the condition.

Check Array Using indexOf()

```
// program to check if an array contains a specified value

const array = ['you', 'will', 'learn', 'javascript'];

const hasValue = array.indexOf('javascript') !== -1;

// check the condition
if(hasValue) {
    console.log('Array contains a value.');
} else {
    console.log('Array does not contain a value.');
}
```

Output

```
Array contains a value.
```

In the above program, the indexOf() method is used with the if...else statement to check if an array contains a specified value.

The indexOf() method searches an array and returns the position of the first occurrence. If the value cannot be found, it returns **-1**.

Note: Both includes() and indexOf() are case sensitive. Hence, **J** and **j** are different.

C) To empty an array

Suppose you have the following array and want to remove all of its elements:

```
let a = [1,2,3];
```
Code language: JavaScript (javascript)

The following shows you several methods to make an array empty.

1) **Assigning it to a new empty array**

This is the fastest way to empty an array:

```
a = [];
```

This code assigned the array a to a new empty array. It works perfectly if you do not have any references to the original array.

See the following example:

```
let b = a;
a = [];
console.log(b); // [1,2,3]
```
Code language: JavaScript (javascript)

In this example, first, the b variable references the array a. Then, the a is assigned to an empty array. The original array still remains unchanged.

2) **Setting its length to zero**

The second way to empty an array is to set its length to zero:

```
a.length = 0;
```

The length property is read/write property of an Array object. When the length property is set to zero, all elements of the array are automatically deleted.

3) **Using splice() method**

The third way to empty an array is to remove all of its elements using the splice() method as shown in the following example:

```
a.splice(0,a.length);
```
Code language: CSS (css)

In this solution, the splice() method removed all the elements of the a array and returned the removed elements as an array.

4) **Using pop() method**

The fourth way to empty an array is to remove each element of the array one by one using the while loop and pop() method:

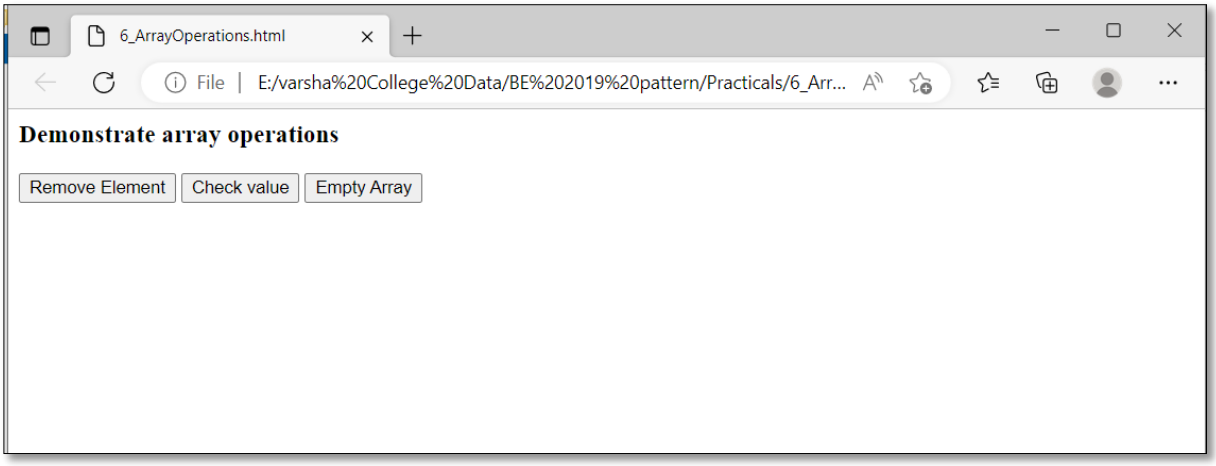
```
while(a.length > 0) {
   a.pop();
}
```

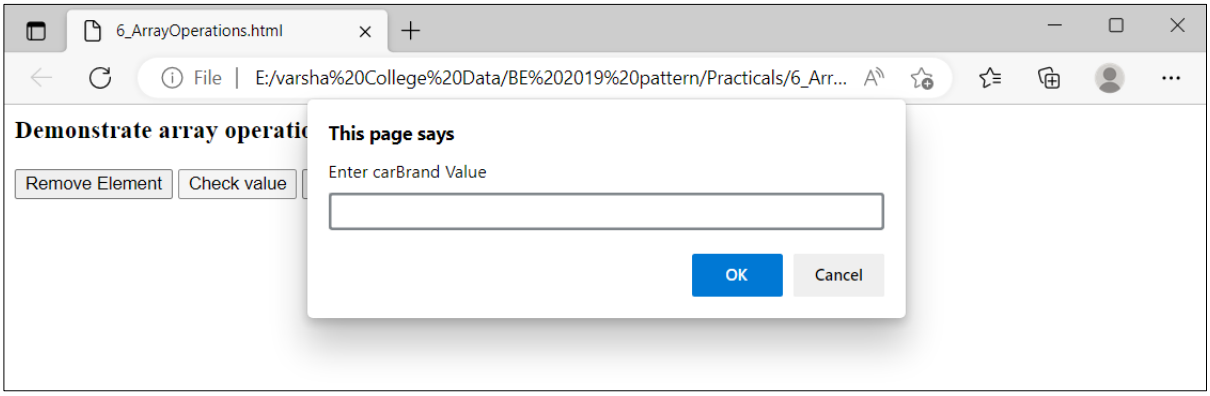
Program:

```
<html>
<body>
<h3>Demonstrate array operations</h3>

<button onclick="removeElement()">Remove Element</button>
<button onclick=" chkValue()">Check value</button>
<button onclick="emptyArray()">Empty Array</button>

<script>



function removeElement() {
   var shoeBrand = ["Nike", " Adidas", " Sparks", " RedTape"];

   console.log("Elements in array before removing: <br>" + shoeBrand );

   // Removing last element from the array
   var poppedElement = shoeBrand.pop();
   console.log("Removed last element from array using pop(): " + poppedElement );

    //display remaining elements present in array after removing
   console.log("New array: " + " " + shoeBrand);

    //removing 2 elemnts from position '1'.
   shoeBrand.splice(1,2);
   console.log("Removed elements  from array using spilce(1,2) : New Array: " + " " +
shoeBrand );
```

```
}

function chkValue(){
  var carBrand = ["Maruti", " BMW", " Kia", " Tata"];
   console.log(carBrand);
    var str1 =   prompt("Enter carBrand Value");
   const hasValue = carBrand.includes(str1)
   console.log(carBrand);
   console.log("Value entered: " + str1);
  //check the condition
   if(hasValue) {
    console.log('Array contains: ' + str1);
    }
    else {
    console.log('Array does not contain:  ' + str1);
          }
}

function emptyArray(){
 var carBrand = ["Maruti", " BMW", " Kia", " Tata"];
  console.log(carBrand);

  carBrand.splice(0,carBrand.length);
//while(a.length > 0) {
  //  a.pop();
  console.log("Empty Array " + " " + carBrand );
}
</script>
</body>
</html>
```

Screenshot's of Output:





Conclusion: