Assignment-2

Q.1 Write a JavaScript program to find all the index positions of a given word within a given string.

```
function findAllIndexes(str, word) {
        let indexes = [];
        let index = -1;

        while ((index = str.indexOf(word, index + 1)) !== -1) {
        indexes.push(index);
        }

        return indexes;
}

let string = "hello world hello";
let word = "world";
let positions = findAllIndexes(string, word);
console.log("The word '" + word + "' appears at positions: " + positions.join(", "));
```

Q.2 Write a JavaScript program to find the first index of a given element in an array using the linear search algorithm.

```
function linearSearch(arr, element) {

for (let i = 0; i < arr.length; i++) {

if (arr[i] === element) {

return i;
}
}
return -1;
}
```

Q.3 Write a JavaScript program to sort a list of elements using Quick sort.

```
function quickSort(arr) {
        if (arr.length <= 1) {
        return arr;
        }

        const pivot = arr[Math.floor(arr.length / 2)];
        const left = [];
        const right = [];
        for (let i = 0; i < arr.length; i++) {
        if (i === Math.floor(arr.length / 2)) {
        continue;
        }
        if (arr[i] < pivot) {
        left.push(arr[i]);
        } else {
        right.push(arr[i]);
        }
        }

        return [...quickSort(left), pivot, ...quickSort(right)];
}

// Example usage:
const arr = [5, 3, 7, 2, 8, 4, 1];
const sortedArr = quickSort(arr);
console.log("Sorted array:", sortedArr);
```

Q.4 Write a JavaScript program to sort a list of elements using Merge sort.

```
function merge_Arrays(left_sub_array, right_sub_array) {
  let array = []
  while (left_sub_array.length && right_sub_array.length) {
        if (left_sub_array[0] < right_sub_array[0]) {
        array.push(left_sub_array.shift())
        } else {
        array.push(right_sub_array.shift())
        }
  }
  return [ ...array, ...left_sub_array, ...right_sub_array ]
}
function merge_sort(unsorted_Array) {
  const middle_index = unsorted_Array.length / 2
  if(unsorted_Array.length < 2) {
        return unsorted_Array
  }
  const left_sub_array = unsorted_Array.splice(0, middle_index)
  return merge_Arrays(merge_sort(left_sub_array),merge_sort(unsorted_Array))
}
unsorted_Array = [39, 28, 44, 4, 10, 83, 11];
console.log("The sorted array will be: ",merge_sort(unsorted_Array));
```

Q.5 Write a JavaScript program to sort a list of elements using Heap sort.

```
function customSort(arr) {
    var N = arr.length;

    for (var i = Math.floor(N / 2) - 1; i >= 0; i--)
    customHeapify(arr, N, i);

    for (var i = N - 1; i > 0; i--) {
    var temp = arr[0];
    arr[0] = arr[i];
    arr[i] = temp;
    customHeapify(arr, i, 0);
    }
}

function customHeapify(arr, N, i) {
    var largest = i;
    var l = 2 * i + 1;
    var r = 2 * i + 2;

    if (l < N && arr[l] > arr[largest])
    largest = l;

    if (r < N && arr[r] > arr[largest])
    largest = r;

    if (largest != i) {
    var swap = arr[i];
    arr[i] = arr[largest];
    arr[largest] = swap;
    customHeapify(arr, N, largest);
    }
}

function customPrintArray(arr) {
    var N = arr.length;
    for (var i = 0; i < N; ++i)
    console.log(arr[i]);
}

var originalArr = [12, 11, 13, 5, 6, 7];
customSort(originalArr);
console.log("Sorted array:" + originalArr);
```

Q.6 Write a JavaScript program to sort a list of elements using Insertion sort.

```
function insertionSort(arr) {
    const n = arr.length;

    for (let i = 1; i < n; i++) {
    let key = arr[i];
    let j = i - 1;

    while (j >= 0 && arr[j] > key) {
    arr[j + 1] = arr[j];
    j--;
    }

    arr[j + 1] = key;
    }

    return arr;
}

// Example usage:
const originalArr = [5, 3, 7, 2, 8, 4, 1];
const sortedArr = insertionSort(originalArr);
console.log("Sorted array:", sortedArr);
```

Q.7 Write a JavaScript program to sort a list of elements using Bubble sort

```
function bubbleSort(arr) {
    const n = arr.length;

    for (let i = 0; i < n - 1; i++) {
    for (let j = 0; j < n - i - 1; j++) {
    if (arr[j] > arr[j + 1]) {
        // Swap arr[j] and arr[j+1]
        let temp = arr[j];
        arr[j] = arr[j + 1];
        arr[j + 1] = temp;
    }
    }
    }

    return arr;
}

// Example usage:
```

```
const arr = [5, 3, 7, 2, 8, 4, 1];
const sortedArr = bubbleSort(arr);
console.log("Sorted array:", sortedArr);
```

Q.8 Write a JavaScript program to sort the characters in a string alphabetically.

```
function sortStringAlphabetically(str) {
        return str.split('').sort().join('');
}

const inputString = "hello world";
const sortedString = sortStringAlphabetically(inputString);
console.log("Original string:", inputString);
console.log("Sorted string:", sortedString);
```

Q.9 Write a JavaScript program to check if a numeric array is sorted or not.

```
function isArraySorted(arr) {
        for (let i = 0; i < arr.length - 1; i++) {
        if (arr[i] > arr[i + 1]) {
        return false;
        }
        }
        return true;
}

// Example usage:
const sortedArray = [1, 2, 3, 4, 5];
const unsortedArray = [5, 3, 7, 2, 8];
console.log("Is sortedArray sorted?", isArraySorted(sortedArray));
console.log("Is unsortedArray sorted?", isArraySorted(unsortedArray));
```

Q.10 Write a JavaScript function to validate whether a given value type is null or not.

```
function isNull(value) {
  return value === null;
}

// Example usage:
console.log(isNull(null)); // true
console.log(isNull(5));    // false
```

Q.11 Write a JavaScript function to validate whether a given value is a number or not.

```
function isNumber(value) {
   return typeof value === 'number' && !isNaN(value);
}

// Example usage:
console.log(isNumber(5));      // true
console.log(isNumber("hello")); // false
```

Q.12 Write a JavaScript function to validate whether a given value is RegExp or not.

```
function isRegExp(value) {
   return Object.prototype.toString.call(value) === '[object RegExp]';
}

// Example usage:
console.log(isRegExp(/test/)); // true
console.log(isRegExp("hello")); // false
```

Q.13 Write a JavaScript program to delete the rollno property from the following object. Also print the object before or after deleting the property.
Sample object:
var student = {
name : "David Rayy",
sclass : "VI",
rollno : 12 };

```
var student = {
         name: "David Rayy",
         sclass: "VI",
         rollno: 12
};

console.log("Object before deleting rollno property:", student);
```

```
delete student.rollno;

console.log("Object after deleting rollno property:", student);
```

Q.14 Write a JavaScript program to display the reading status (i.e. display book name, author name and reading status) of the following books.

```
var library = [
 {
 author: 'Bill Gates',
 title: 'The Road Ahead',
 readingStatus: true
 },
 {
 author: 'Steve Jobs',
 title: 'Walter Isaacson',
 readingStatus: true
 },
 {
 author: 'Suzanne Collins',
 title: 'Mockingjay: The Final Book of The Hunger Games',
 readingStatus: false
}];
```

```
var library = [
        {
        author: 'Bill Gates',
        title: 'The Road Ahead',
        readingStatus: true
        },
        {
        author: 'Steve Jobs',
        title: 'Walter Isaacson',
        readingStatus: true
        },
        {
        author: 'Suzanne Collins',
        title: 'Mockingjay: The Final Book of The Hunger Games',
        readingStatus: false
        }
];

for (var i = 0; i < library.length; i++) {
        var book = library[i];
```

Assignment-2

Q.1 Write a JavaScript program to find all the index positions of a given word within a given string.

```
function findAllIndexes(str, word) {
        let indexes = [];
        let index = -1;

        while ((index = str.indexOf(word, index + 1)) !== -1) {
        indexes.push(index);
        }

        return indexes;
}

let string = "hello world hello";
let word = "world";
let positions = findAllIndexes(string, word);
console.log("The word '" + word + "' appears at positions: " + positions.join(", "));
```

Q.2 Write a JavaScript program to find the first index of a given element in an array using the linear search algorithm.

```
function linearSearch(arr, element) {

for (let i = 0; i < arr.length; i++) {

if (arr[i] === element) {

return i;
}
}
return -1;
}
```

Q.3 Write a JavaScript program to sort a list of elements using Quick sort.

```
function quickSort(arr) {
        if (arr.length <= 1) {
        return arr;
        }

        const pivot = arr[Math.floor(arr.length / 2)];
        const left = [];
        const right = [];
        for (let i = 0; i < arr.length; i++) {
        if (i === Math.floor(arr.length / 2)) {
        continue;
        }
        if (arr[i] < pivot) {
        left.push(arr[i]);
        } else {
        right.push(arr[i]);
        }
        }

        return [...quickSort(left), pivot, ...quickSort(right)];
}

// Example usage:
const arr = [5, 3, 7, 2, 8, 4, 1];
const sortedArr = quickSort(arr);
console.log("Sorted array:", sortedArr);
```

Q.4 Write a JavaScript program to sort a list of elements using Merge sort.

```
function merge_Arrays(left_sub_array, right_sub_array) {
  let array = []
  while (left_sub_array.length && right_sub_array.length) {
        if (left_sub_array[0] < right_sub_array[0]) {
        array.push(left_sub_array.shift())
        } else {
        array.push(right_sub_array.shift())
        }
  }
  return [ ...array, ...left_sub_array, ...right_sub_array ]
}
function merge_sort(unsorted_Array) {
  const middle_index = unsorted_Array.length / 2
  if(unsorted_Array.length < 2) {
        return unsorted_Array
  }
  const left_sub_array = unsorted_Array.splice(0, middle_index)
  return merge_Arrays(merge_sort(left_sub_array),merge_sort(unsorted_Array))
}
unsorted_Array = [39, 28, 44, 4, 10, 83, 11];
console.log("The sorted array will be: ",merge_sort(unsorted_Array));
```

Q.5 Write a JavaScript program to sort a list of elements using Heap sort.

```
function customSort(arr) {
        var N = arr.length;

        for (var i = Math.floor(N / 2) - 1; i >= 0; i--)
        customHeapify(arr, N, i);

        for (var i = N - 1; i > 0; i--) {
        var temp = arr[0];
        arr[0] = arr[i];
        arr[i] = temp;
        customHeapify(arr, i, 0);
        }
}

function customHeapify(arr, N, i) {
        var largest = i;
        var l = 2 * i + 1;
        var r = 2 * i + 2;

        if (l < N && arr[l] > arr[largest])
        largest = l;

        if (r < N && arr[r] > arr[largest])
        largest = r;

        if (largest != i) {
        var swap = arr[i];
        arr[i] = arr[largest];
        arr[largest] = swap;
        customHeapify(arr, N, largest);
        }
}

function customPrintArray(arr) {
        var N = arr.length;
        for (var i = 0; i < N; ++i)
        console.log(arr[i]);
}

var originalArr = [12, 11, 13, 5, 6, 7];
customSort(originalArr);
console.log("Sorted array:" + originalArr);
```