JavaScript Assignment

Abhrajyoti Pal 1NT17IS119

All code in this document are pushed to my GitHub Repository:
https://github.com/Meerkats1999/webtech-lab

Access the folder named **assignment-1**

Home page: https://github.com/Meerkats1999/webtech-lab/blob/master/assignment-1/home.html

1. Write a function that takes in an array of integers and returns the integers that are either **palindromes** or **almost-palindromes**. An **almost-palindrome** is any integer that can be rearranged to form a palindrome.

For example, the numbers `677` and `338` are both **almost-palindromes**, since they can be rearranged to form `767` and `383`, respectively.

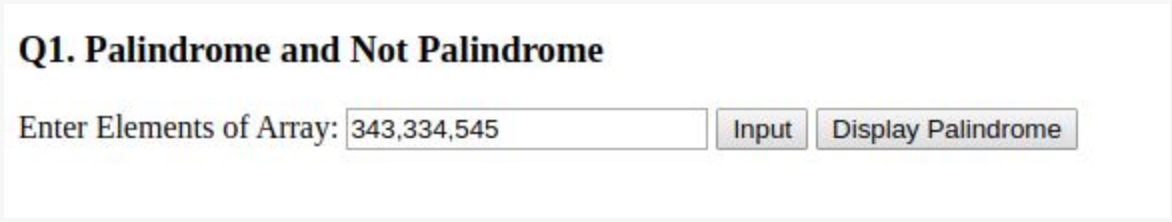
https://github.com/Meerkats1999/webtech-lab/blob/master/assignment-1/javascript/palindrome.js

```
var numArray = [];

var palArray = [];

var notPalArray = [];



function fetchDataPalindrome() {

   var value = document.getElementById("palNum").value;

   numArray = value.split(",");

}



function isPalindrome(number) {

   var l = 0;
```

```
    var h = number.length - 1;


    while (h > l)
        if (number[l++] != number[h--])
            return false;


    return true;
}


function isAlmostPalindrome(number) {
    var n = number.length;
    for (i = 0; i < n - 1; i++) {
        var str1 = number.substr(i + 1, n - i - 1);
        var str2 = number.substr(0, i + 1);


        if (isPalindrome(str1.concat(str2)))
            return true
    }
    return false;
}


function displayPalindrome() {
```

```
    len = numArray.length;


    //Checks for Palindrome
    for (i = 0; i < len; i++) {
        if (isPalindrome(numArray[i])) {
            palArray.push(numArray[i])
        }
        else {
            notPalArray.push(numArray[i])
        }
    }


    // Checks for Almost Palindrome
    if (notPalArray.length != 0) {
        for (i = 0; i < notPalArray.length; i++) {
            if (isAlmostPalindrome(notPalArray[i])) {
                palArray.push(notPalArray[i])
            }
        }
    }


    alert(palArray)
}
```

Output:



2. Create a function that takes an array and return the most frequently occuring element contained within it.

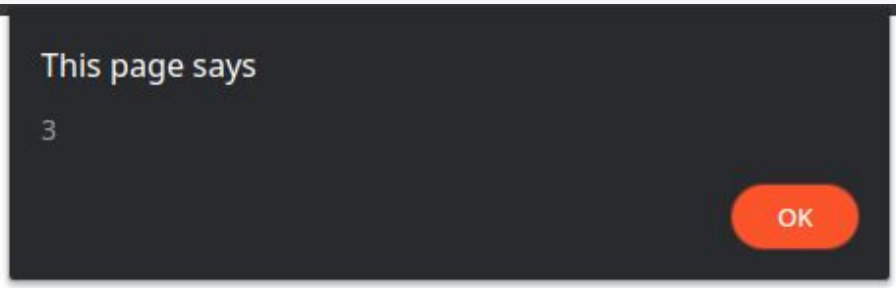
https://github.com/Meerkats1999/webtech-lab/blob/master/assignment-1/javascript/frequentItem.js

```
var numArray = [];



function fetchDataFrequent() {

   var value = document.getElementById("frequentNum").value;

   numArray = value.split(",");

}



function displayFrequent() {
```

```
numArray.sort()

n = numArray.sort()


var max_count = 1, res = numArray[0], curr_count = 1;


for (i = 1; i < n; i++) {

    if (numArray[i] == numArray[i - 1])

        curr_count++;

    else {

        if (curr_count > max_count) {

            max_count = curr_count;

            res = numArray[i - 1];

        }

        curr_count = 1;

    }

}


if (curr_count > max_count) {

    max_count = curr_count;

    res = numArray[n - 1];

}


alert(res)
```

```
}
```

Output:



3. This robot roams around a 2D grid. It starts at (0, 0) facing North. After each time it moves, the robot rotates 90 degrees clockwise. Given the amount the robot has moved each time, you have to calculate the robot's final position.

https://github.com/Meerkats1999/webtech-lab/blob/master/assignment-1/javascript/robot.js

```
function displayFinalPosition() {



   var commands = [];

   var value = document.getElementById("commands").value;

   commands = value.split(',').map(function (i) {

       return parseInt(i, 10);

   })
```

```
var n = commands.length;

var count = 1;

var x = y = 0;



for (i = 0; i < n; i++) {

    if (i + 1 == Math.pow(count, 2) - 2 * count + 4) {

        count++;

        if (i % 2 == 0) {

            x -= commands[i];

        }

        else {

            y -= commands[i];

        }

    }

    else {

        if (i % 2 == 0) {

            x += commands[i];

        }

        else {

            y += commands[i];

        }

    }
```

```
    }


    alert(`${x},${y}`);
}
```

Output:

4. An input string can be completed if additional letters can be added and no letters need to be taken away to match the word. Furthermore, the order of the letters in the input string must be the same as the order of letters in the final word.

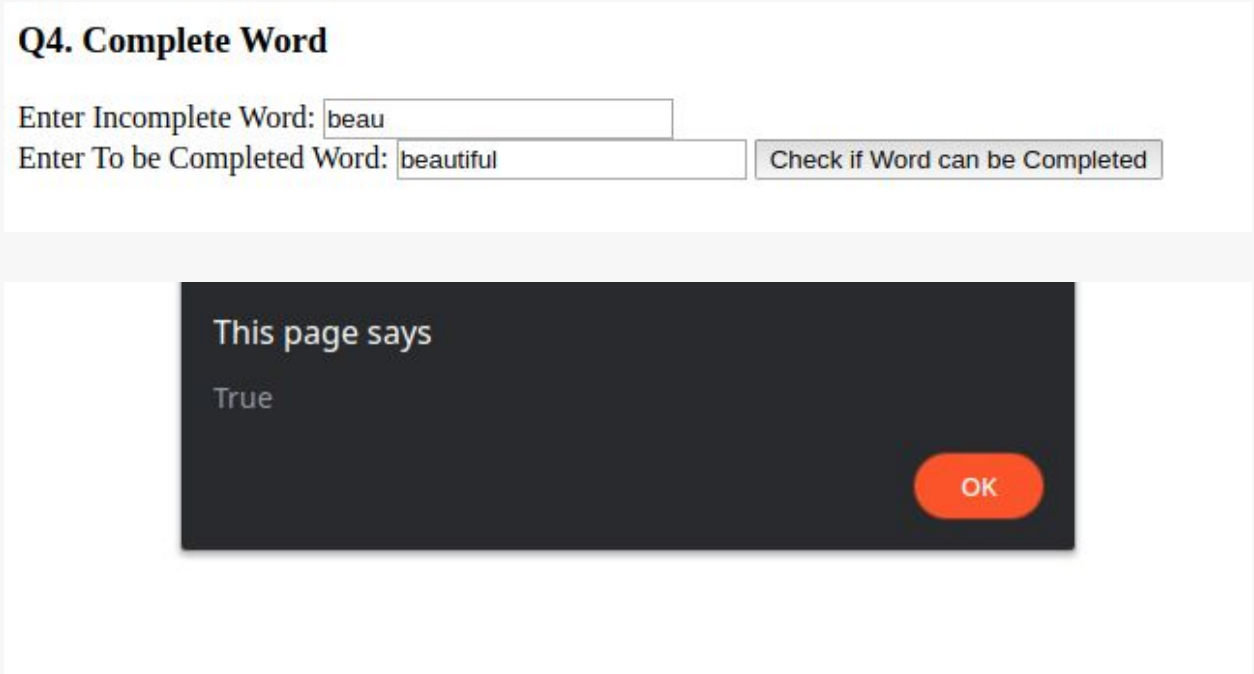
Create a function that, given an input string, determines if the word can be completed.

https://github.com/Meerkats1999/webtech-lab/blob/master/assignment-1/javascript/completeWord.js

```
function completeWord() {
```

```
    var input1 = document.getElementById("word1").value;
    var input2 = document.getElementById("word2").value;
    var word1 = input1.split('');
    var word2 = input2.split('');


    var pointer = 0;
    for (i = 0; i < word2.length; i++) {
        var index = word1.indexOf(word2[i])
        if (index > -1) {
            if (index > 0) {
                pointer = 1;
            }
            word1.splice(index, 1)
        }
    }


    if (word1.length > 0 || pointer == 1) {
        alert('False')
    } else {
        alert('True')
    }
}
```

Output:



5. You are playing a game of JavaScript & Jackalopes with your friends and need to roll dice as part of the game. None of you actually own dice, but you do have a computer handy!

You'll be given a string representing the number of dice to roll, how many faces each die has, and a "modifier" to apply to the final result after adding up all the dice. For example, rolling a single six-sided die with no modifier might be represented by the string `"1d6"` — one die with six sides and values ranging from 1 through 6. If you wanted to add 2 to the result of rolling the same die, you might represent that as `"1d6+2"`.

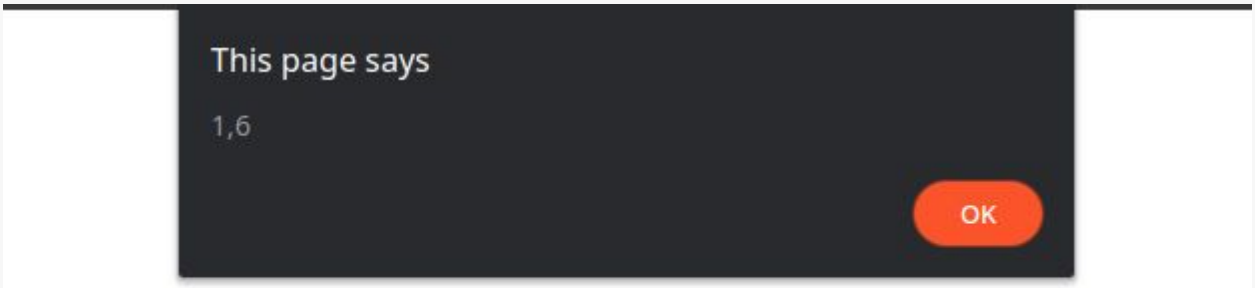
Create a function that takes a string representing a set of dice to be rolled as an argument, and returns an array of two numbers representing the minimum and maximum possible values that could be achieved.

https://github.com/Meerkats1999/webtech-lab/blob/master/assignment-1/javascript/dice.js

```
function Dice() {

   var diceRange = document.getElementById("diceRange").value.split('d')

   alert(diceRange)
```

```
    var initialVal, finalVal;

    var modVal = diceRange[1].slice(1, 3)

    var numDice = diceRange[0];



    if (numDice == "") {

        initialVal = eval(1 + modVal)

    }

    else {

        initialVal = eval(numDice + modVal)

    }



    if (numDice > 1) {

        finalVal = eval(numDice * 6 + modVal)

    } else {

        if (numDice == 0) {

            finalVal = eval(numDice + modVal)

        } else {

            finalVal = eval(diceRange[1])

        }

    }


}
```

Output:



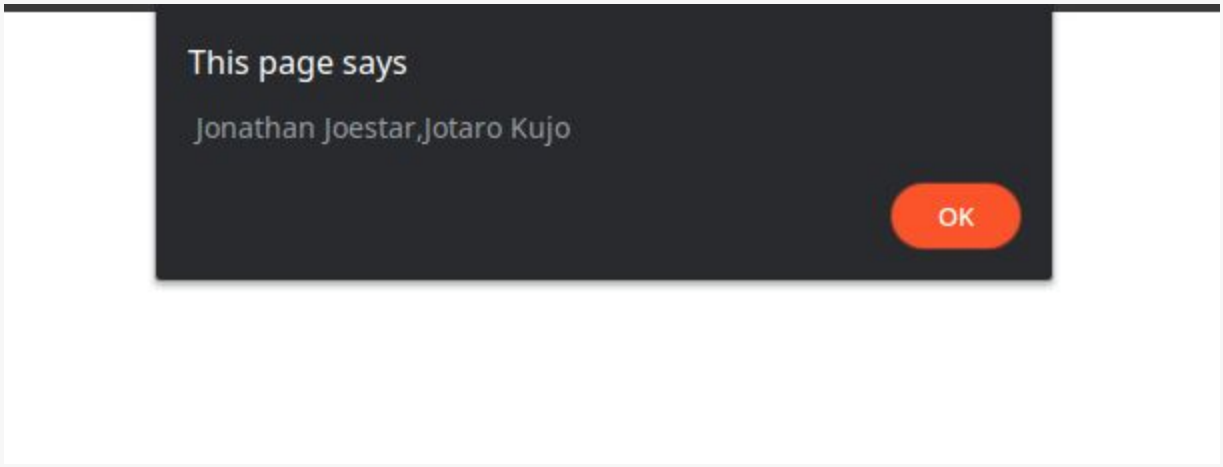
6. Write a sorting function that takes in an array of names and sorts them **by last name** either alphabetically (ASC) or reverse-alphabetically (DESC).

https://github.com/Meerkats1999/webtech-lab/blob/master/assignment-1/javascript/lastname.js

```
function lastNameSort() {

   var names = document.getElementById('name').value.split(',');

   var order = document.getElementById('order').value;



   if(order == "ASC"){

       var sorted = names.sort(compare);

   }

   else{
```

```
        var sorted = names.sort(compare).reverse();
    }
    alert(sorted)
}



function compare(a, b) {
    var splitA = a.split(" ");
    var splitB = b.split(" ");
    var lastA = splitA[splitA.length - 1];
    var lastB = splitB[splitB.length - 1];

    if (lastA < lastB) return -1;
    if (lastA > lastB) return 1;
    return 0;
}
```

Output:



7. Practice Regular Expressions Concepts

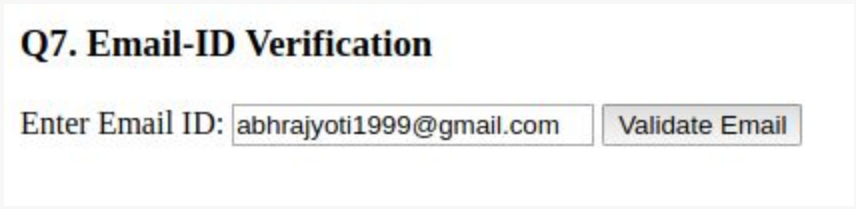
Email iD verification using regular expression. Need to accept valid Email ID.

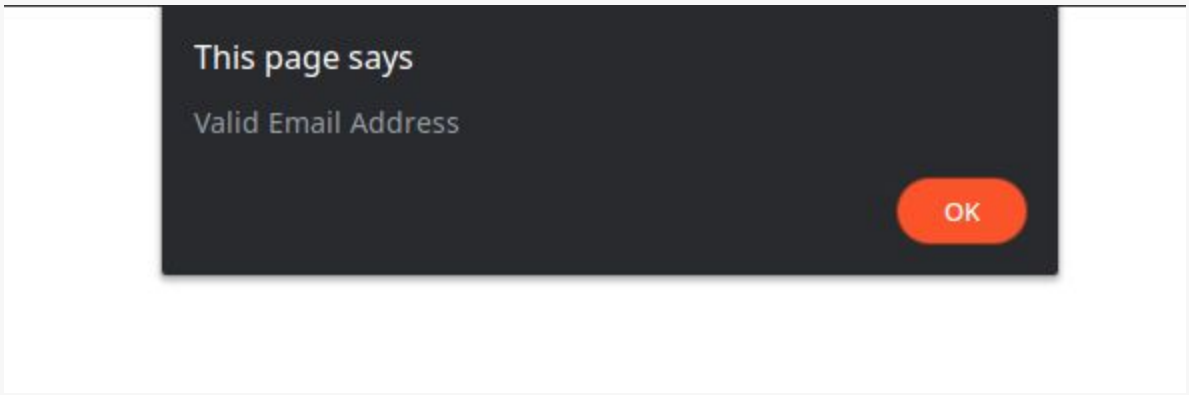
https://github.com/Meerkats1999/webtech-lab/blob/master/assignment-1/javascript/emailValidation.js

```
function validateEmail() {

   email = document.getElementById("email").value;

   email = email.toString()



   var filter = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}$/;



   if (filter.test(email.toLowerCase()) == false) {

       alert('Not Valid Email Address');

   }

   else {

       alert('Valid Email Address')
```

```
    }

}
```

Output:





8. Create a function that takes in an array of grass heights and a **variable** sequence of lawn mower cuts and outputs the array of successive grass heights.

If **after a cut**, any single element in the array reaches zero or negative, return "Done" instead of the array of new heights.

https://github.com/Meerkats1999/webtech-lab/blob/master/assignment-1/javascript/lawnmower.js

```
function finalHeight() {

   var grassHeights =
document.getElementById("grassHeight").value.split(',').map(function (i) {
```

```
        return parseInt(i, 10);

    })

    var cutHeights =
document.getElementById("cutHeight").value.split(',').map(function (i) {

        return parseInt(i, 10);

    })

    var tempHeights = grassHeights

    var flag = 0;



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

        for(j=0;j<grassHeights.length;j++){

            grassHeights[j] = grassHeights[j] - cutHeights[i];

            if(grassHeights[j]==0 || grassHeights[j]<0){

                flag = 1;

                break;

            }

        }

        if(flag==1){

            grassHeights = tempHeights;

            break;

        }

        else{

            tempHeights = grassHeights;

        }
```

```
    }


    alert(grassHeights)
}
```

Output: