Basic Algorithm Scripting: Reverse a String (arr.reverse())

reverseString("hello")should become "olleh".

reverseString("Howdy")should become "ydwoH".

reverseString("Greetings from Earth")should return "htraE morf sgniteerG".

function reverseString(str) {

var splited\_array = str.split('');

var reversed\_array = splited\_array.reverse()

var str = reversed\_array.join('')

return str;

}

reverseString("hello");

## Basic Algorithm Scripting: Factorialize a Number

Return the factorial of the provided integer.

If the integer is represented with the letter n, a factorial is the product of all positive integers less than or equal to n.

Factorials are often represented with the shorthand notation n!

For example: 5! = 1 \* 2 \* 3 \* 4 \* 5 = 120

function factorialize(num) {

if (num === 1 || num === 0) {

return 1;

}

for (var i = num - 1; i > 0; i--) {

num \*= i;

}

return num;

}

factorialize(5);

## Basic Algorithm Scripting: Find the Longest Word in a String

findLongestWordLength("The quick brown fox jumped over the lazy dog")should return a number.

Passed

findLongestWordLength("The quick brown fox jumped over the lazy dog")should return 6.

Passed

findLongestWordLength("May the force be with you")should return 5.

Passed

findLongestWordLength("Google do a barrel roll")should return 6.

Passed

findLongestWordLength("What is the average airspeed velocity of an unladen swallow")should return 8.

Passed

findLongestWordLength("What if we try a super-long word such as otorhinolaryngology")should return 19.

|  |  |
| --- | --- |
|  | function findLongestWord(str) { |
|  | var strSplit = str.split(' '); |
|  | var longestWord = 0; |
|  | for(var i = 0; i < strSplit.length; i++){ |
|  | if(strSplit[i].length > longestWord){ |
|  | longestWord = strSplit[i].length; |
|  | } |
|  | } |
|  | return longestWord; |
|  | } |
|  | findLongestWord("The quick brown fox jumped over the lazy dog"); |

Or

|  |
| --- |
| function findLongestWord(str) { |
|  | var longestWord = str.split(' ').sort(function(a, b) { return b.length - a.length; }); |
|  | return longestWord[0].length; |
|  | } |
|  | findLongestWord("The quick brown fox jumped over the lazy dog"); |

## Basic Algorithm Scripting: Return Largest Numbers in Arrays

largestOfFour([[4, 5, 1, 3], [13, 27, 18, 26], [32, 35, 37, 39], [1000, 1001, 857, 1]])should return an array.

largestOfFour([[13, 27, 18, 26], [4, 5, 1, 3], [32, 35, 37, 39], [1000, 1001, 857, 1]])should return [27, 5, 39, 1001].

largestOfFour([[4, 9, 1, 3], [13, 35, 18, 26], [32, 35, 97, 39], [1000000, 1001, 857, 1]])should return [9, 35, 97, 1000000].

largestOfFour([[17, 23, 25, 12], [25, 7, 34, 48], [4, -10, 18, 21], [-72, -3, -17, -10]])should return [25, 48, 21, -3].

function largestOfFour(arr) {

var results = [];

for (var n = 0; n < arr.length; n++) {

var largestNumber = arr[n][0];

for (var sb = 1; sb < arr[n].length; sb++) {

if (arr[n][sb] > largestNumber) {

largestNumber = arr[n][sb];

}

}

results[n] = largestNumber;

}

return results;

}

## Basic Algorithm Scripting: Confirm the Ending

confirmEnding("Congratulation", "on")should return true.

confirmEnding("Connor", "n")should return false.

function confirmEnding(str, target) {

var tgLength = target.length;

var strLength = str.length;

var newString = str.substring(strLength - tgLength)

var returnValue = false;

if(newString == target) {

returnValue = true;

} else {returnValue = false}

return returnValue;

}

confirmEnding("Connor", "n");

\*\*\* substring

var anyString = 'Mozilla';

// Displays 'M'

console.log(anyString.substring(0, 1));

console.log(anyString.substring(1, 0));

// Displays 'Mozill'

console.log(anyString.substring(0, 6));

// Displays 'lla'

console.log(anyString.substring(4));

console.log(anyString.substring(4, 7));

console.log(anyString.substring(7, 4));

// Displays 'Mozilla'

console.log(anyString.substring(0, 7));

console.log(anyString.substring(0, 10));

// Displays 'illa' the last 4 characters

var anyString = 'Mozilla';

var anyString4 = anyString.substring(anyString.length - 4);

console.log(anyString4);

// Displays 'zilla' the last 5 characters

var anyString = 'Mozilla';

var anyString5 = anyString.substring(anyString.length - 5);

console.log(anyString5);

or use Slice (대체함 https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref\_slice)

function confirmEnding(str, target) {

return str.slice(str.length - target.length) === target;

}

confirmEnding("He has to give me a new name", "name");

## Basic Algorithm Scripting: Truncate a String

Truncate a string (first argument) if it is longer than the given maximum string length (second argument). Return the truncated string with a ...ending.

truncateString("Peter Piper picked a peck of pickled peppers", 11)should return "Peter Piper...".

Passed

truncateString("A-tisket a-tasket A green and yellow basket", "A-tisket a-tasket A green and yellow basket".length)should return "A-tisket a-tasket A green and yellow basket".

function truncateString(str, num) {

var slicedStr = str.slice(0, num)

var strLength = str.length;

var finalString = ""

if(str.length > num) {

finalString = slicedStr + "..."

} else {finalString = slicedStr}

return finalString;

}

truncateString("A-tisket a-tasket A green and yellow basket", 8);

## Basic Algorithm Scripting: Finders Keepers

findElement([1, 3, 5, 8, 9, 10], function(num) { return num % 2 === 0; })should return 8.

Passed

findElement([1, 3, 5, 9], function(num) { return num % 2 === 0; })should return undefined.

function findElement(arr, func) {

let num = 0;

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

num = arr[i];

if (func(num)) {

return num;

}

}

return undefined;

}

findElement([1, 2, 3, 4], num => num % 2 === 0);

## Basic Algorithm Scripting: Boo who

Check if a value is classified as a boolean primitive. Return true or false.

Boolean primitives are true and false.

function booWho(bool) {

// What is the new fad diet for ghost developers? The Boolean.

if (bool === true || bool === false) {

return true;

} else return false

}

booWho(null);

or use typeof

function booWho(bool) {

return typeof bool === 'boolean';

}

// test here

booWho(null);

## Basic Algorithm Scripting: Title Case a Sentence

titleCase("I'm a little tea pot")should return I'm A Little Tea Pot.

titleCase("sHoRt AnD sToUt")should return Short And Stout.

function titleCase(str) {

var convertToArray = str.toLowerCase().split(" ");

var result = convertToArray.map(function(val){

return val.replace(val.charAt(0), val.charAt(0).toUpperCase());

});

return result.join(" ");

}

titleCase("I'm a little tea pot");

## Basic Algorithm Scripting: Slice and Splice

frankenSplice([1, 2, 3], [4, 5], 1)should return [4, 1, 2, 3, 5].

frankenSplice([1, 2], ["a", "b"], 1)should return ["a", 1, 2, "b"].

frankenSplice(["claw", "tentacle"], ["head", "shoulders", "knees", "toes"], 2)should return ["head", "shoulders", "claw", "tentacle", "knees", "toes"].

function frankenSplice(arr1, arr2, n) {

// It's alive. It's alive!

let localArray = arr2.slice();

for (let i = 0; i < arr1.length; i++) {

localArray.splice(n, 0, arr1[i]);

n++;

}

return localArray;

}

frankenSplice([1, 2, 3], [4, 5], 1);

## Basic Algorithm Scripting: Falsy Bouncer

Remove all falsy values from an array.

Falsy values in JavaScript are false, null, 0, "", undefined, and NaN.

bouncer([false, null, 0, NaN, undefined, ""])should return [].

bouncer([1, null, NaN, 2, undefined])should return [1, 2].

function bouncer(arr) {

return arr.filter(Boolean);

}

bouncer([7, "ate", "", false, 9]);

## Basic Algorithm Scripting: Where do I Belong

getIndexToIns([10, 20, 30, 40, 50], 35)should return 3.

getIndexToIns([10, 20, 30, 40, 50], 35)should return a number.

getIndexToIns([10, 20, 30, 40, 50], 30)should return 2.

getIndexToIns([10, 20, 30, 40, 50], 30)should return a number.

getIndexToIns([40, 60], 50)should return 1.

getIndexToIns([40, 60], 50)should return a number.

getIndexToIns([3, 10, 5], 3)should return 0.

getIndexToIns([3, 10, 5], 3)should return a number.

getIndexToIns([5, 3, 20, 3], 5)should return 2.

getIndexToIns([5, 3, 20, 3], 5)should return a number.

getIndexToIns([2, 20, 10], 19)should return 2.

getIndexToIns([2, 20, 10], 19)should return a number.

getIndexToIns([2, 5, 10], 15)should return 3.

getIndexToIns([2, 5, 10], 15)should return a number.

getIndexToIns([], 1)should return 0.

getIndexToIns([], 1)should return a number.

function getIndexToIns(arr, num) {

arr.sort(function(a, b) {

return a - b;

});

for (var a = 0; a < arr.length; a++) {

if (arr[a] >= num)

return a;

}

return arr.length;

}

## Basic Algorithm Scripting: Mutations

mutation(["hello", "hey"])should return false.

mutation(["hello", "Hello"])should return true.

mutation(["zyxwvutsrqponmlkjihgfedcba", "qrstu"])should return true.

mutation(["Mary", "Army"])should return true.

mutation(["Mary", "Aarmy"])should return true.

mutation(["Alien", "line"])should return true.

mutation(["floor", "for"])should return true.

mutation(["hello", "neo"])should return false.

mutation(["voodoo", "no"])should return false.

function mutation(arr) {

var test = arr[1].toLowerCase();

var target = arr[0].toLowerCase();

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

if (target.indexOf(test[i]) < 0)

return false;

}

return true;

}

mutation(["hello", "hey"])

## Basic Algorithm Scripting: Chunky Monkey

chunkArrayInGroups(["a", "b", "c", "d"], 2)should return [["a", "b"], ["c", "d"]].

chunkArrayInGroups([0, 1, 2, 3, 4, 5], 3)should return [[0, 1, 2], [3, 4, 5]].

chunkArrayInGroups([0, 1, 2, 3, 4, 5], 2)should return [[0, 1], [2, 3], [4, 5]].

chunkArrayInGroups([0, 1, 2, 3, 4, 5], 4)should return [[0, 1, 2, 3], [4, 5]].

chunkArrayInGroups([0, 1, 2, 3, 4, 5, 6], 3)should return [[0, 1, 2], [3, 4, 5], [6]].

chunkArrayInGroups([0, 1, 2, 3, 4, 5, 6, 7, 8], 4)should return [[0, 1, 2, 3], [4, 5, 6, 7], [8]].

chunkArrayInGroups([0, 1, 2, 3, 4, 5, 6, 7, 8], 2)should return [[0, 1], [2, 3], [4, 5], [6, 7], [8]].

function chunkArrayInGroups(arr, size) {

var temp = [];

var result = [];

for (var a = 0; a < arr.length; a++) {

if (a % size !== size - 1)

temp.push(arr[a]);

else {

temp.push(arr[a]);

result.push(temp);

temp = [];

}

}

if (temp.length !== 0)

result.push(temp);

return result;

}

Or

function chunkArrayInGroups(arr, size) {

// Break it up.

var arr2 = [];

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

arr2.push(arr.slice(i , i+size));

}

return arr2;

}

아.. 모르겠다 ;;