**Base-001**

Дана строка из символов латинского алфавита. Написать функцию transStr(str),

которая строчные гласные буквы делает прописными, а прописные – строчными.

Функция должна возвратить результат. Например :

console.log(transStr(‘colOr’)); // ‘cOlor’

console.log(transStr(‘sOrrY’)); // ‘sorry’

Решение :

function transStr(str) {  
 str = str || '';  
 let i,  
 len = str.length,  
 char,  
 result = '',  
 obj = {  
 a : 'A',  
 e : 'E',  
 o : 'O',  
 i : 'I',  
 u : 'U',  
 y : 'Y'  
 };  
  
 for (i = 0; i < len; i++) {  
 char = str[i].toLowerCase();  
 if (char in obj) {  
 result +=

(obj[char] === str[i]) ? str[i].toLowerCase() : str[i].toUpperCase();  
 }else{  
 result += str[i];  
 }  
 }  
 return result;  
}  
  
/\* =================== the testing function ======================= \*/  
function test (testedFunc, result, ...data) {  
 testedFunc(...data) === result ?  
 console.log(`+ + + + + + + : ${data} --> ${result}`) :  
 console.log(`- - - - - - - : ${data} --> ${result}`);  
}

/\* =================== tests ======================= \*/  
test(transStr, '', '');  
test(transStr, 'colOr', 'cOlor');  
test(transStr, ' cOOOOr alisA ', ' coooor AlIsa ');  
test(transStr, 'auaueoiuaueye', 'AUAUEOIUAUEYE');  
test(transStr, 'dfrtfdrfggtrfd', 'dfrtfdrfggtrfd');  
test(transStr, '^^^\*&%$###$#!@#@!~~', '^^^\*&%$###$#!@#@!~~');  
test(transStr, '', undefined);  
test(transStr, '', 12345);  
test(transStr, 'a', 'A', 'O');

**Base-002**

<https://www.codewars.com/kata/sum-of-digits-slash-digital-root>

In this kata, you must create a digital root function.

A digital root is the recursive sum of all the digits in a number. Given n, take the sum of the digits of n. If that value has two digits, continue reducing in this way until a single-digit number is produced. This is only applicable to the natural numbers.

Here's how it works:

digitalRoot(16)

=> 1 + 6

=> 7

digitalRoot(942)

=> 9 + 4 + 2

=> 15 ...

=> 1 + 5

=> 6

digitalRoot(132189)

=> 1 + 3 + 2 + 1 + 8 + 9

=> 24 ...

=> 2 + 4

=> 6

digitalRoot(493193)

=> 4 + 9 + 3 + 1 + 9 + 3

=> 29 ...

=> 2 + 9

=> 11 ...

=> 1 + 1

=> 2

function digitalRoot(n) {  
 while (n >= 10) {  
 let i,  
 result = 0,  
 str = n + '',  
 len = str.length;  
 for (i = 0; i < len; i++) {  
 result += +str[i];  
 }  
 n = result;  
 }  
 return n;  
}  
  
/\* =================== the testing function ======================= \*/  
function test (testedFunc, result, ...data) {  
 testedFunc(...data) === result ?  
 console.log(`+ + + + + + + : ${data} --> ${result}`) :  
 console.log(`- - - - - - - : ${data} --> ${result}`);  
}  
  
/\* =================== tests ======================= \*/  
test(digitalRoot, 7, 16);  
test(digitalRoot, 6, 942);  
test(digitalRoot, 6, 132189);  
test(digitalRoot, 2, 493193);

**Base-003**

https://www.codewars.com/kata/ones-and-zeros

Given an array of one's and zero's convert the equivalent binary value to an integer.

Eg: [0, 0, 0, 1] is treated as 0001 which is the binary representation of 1.

Examples:

Testing: [0, 0, 0, 1] ==> 1

Testing: [0, 0, 1, 0] ==> 2

Testing: [0, 1, 0, 1] ==> 5

Testing: [1, 0, 0, 1] ==> 9

Testing: [0, 0, 1, 0] ==> 2

Testing: [0, 1, 1, 0] ==> 6

Testing: [1, 1, 1, 1] ==> 15

Testing: [1, 0, 1, 1] ==> 11

However, the arrays can have varying lengths, not just limited to 4.

const binaryArrayToNumber = arr =>  
 arr  
 .reverse()  
 .reduce((a, b, i) => a + b \* Math.pow(2, i));

/\* =================== the testing function ======================= \*/  
function test (testedFunc, result, ...data) {  
 testedFunc(...data) === result ?  
 console.log(`+ + + + + + + : ${data} --> ${result}`) :  
 console.log(`- - - - - - - : ${data} --> ${result}`);  
}  
  
/\* =================== tests ======================= \*/  
test(binaryArrayToNumber, 1, [0,0,0,1]);  
test(binaryArrayToNumber, 2, [0,0,1,0]);  
test(binaryArrayToNumber, 15, [1,1,1,1]);  
test(binaryArrayToNumber, 6, [0,1,1,0]);

**Base-004**

https://www.codewars.com/kata/mexican-wave/train/javascript

Task

In this simple Kata your task is to create a function that turns a string into a Mexican Wave. You will be passed a string and you must return that string in an array where an uppercase letter is a person standing up.

Rules

1. The input string will always be lower case but maybe empty.

2. If the character in the string is whitespace then pass over it as if it was an empty seat.

Example

wave('hello') => ['Hello', 'hEllo', 'heLlo', 'helLo', 'hellO']

function wave(s){  
 let arr = [];  
 let str = s.toLowerCase();  
 str.split('').forEach(function(elem, i) {  
 if (elem !== ' ') {  
 arr.push(str.slice(0, i) + str[i].toUpperCase() + str.slice(i+1));  
 }  
 })  
 return arr;  
}

/\* =================== the testing function ======================= \*/  
function test (testedFunc, result, data) {  
 testedFunc(data).join(',') === result ?  
 console.log(`+ + + + + + + : ${data} --> ${result}`) :  
 console.log(`- - - - - - - : ${data} --> ${result}`);  
}  
  
/\* =================== tests ======================= \*/  
test(wave, 'Hello,hEllo,heLlo,helLo,hellO', 'hello');  
test(wave, 'Codewars,cOdewars,coDewars,codEwars,codeWars,codewArs,codewaRs,codewarS', 'codewars');  
test(wave, ' Gap , gAp , gaP ', ' gap ');