Problem 1:

Take 2 strings s1 and s2 including only letters from ato z. Return a new sorted string, the longest possible, containing distinct letters,

each taken only once - coming from s1 or s2. #Examples: ``` a = "xyaabbbccccdefww" b = "xxxxyyyyabklmopq" longest(a, b) -> "abcdefklmopqwxy"

a = "abcdefghijklmnopqrstuvwxyz" longest(a, a) -> "abcdefghijklmnopqrstuvwxyz" ```

Ans: function longest(s1, s2) {

var result = "";

for (var i=0; i<s1.length; i++) {

if( result.indexOf(s1[i]) === -1) {

result += s1[i];

}

}

for (var i=0; i<s2.length; i++) {

if( result.indexOf(s2[i]) === -1) {

result += s2[i];

}

}

return result.split('').sort().join('');

}

Problem 2:

You are given an array (which will have a length of at least 3, but could be very large) containing integers. The array is either entirely comprised of odd integers or entirely comprised of even integers except for a single integer N. Write a method that takes the array as an argument and returns this "outlier" N.

Examples

[2, 4, 0, 100, 4, 11, 2602, 36]

Should return: 11 (the only odd number)

[160, 3, 1719, 19, 11, 13, -21]

Should return: 160 (the only even number)

Ans:

function findOutlier(integers){

//your code here

var evens = [];

var odds = [];

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

{

if ((integers[i] % 2) == 0)

{

evens.push(integers[i]);

}

else

{

odds.push(integers[i]);

}

}

var elen = evens.length;

var olen = odds.length;

if (elen > olen)

{

return odds[0];

}

else

{

return evens[0];

}

}

Problem 3:

You are given an array strarr of strings and an integer k. Your task is to return the first longest string consisting of k consecutive strings taken in the array.

Ans:

function longestConsec(strarr, k) {

// your code

var string = ""

var n = strarr.length;

if (n === 0 || k > n || k <= 0) {

return string;

}

else {

//find out which word is the longest word

//return "that word + (k-1)after that word:

var lengthWord = [];

var max = 0;

var longest\_word = ""

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

if (strarr[i].length > max ) {

max = strarr[i].length;

longest\_word = strarr[i]

if (i === strarr.length -1 ) {

return longest\_word;

}

}

for (var j = i + 1; j <= k; j++) {

string = longest\_word.concat(strarr[j])

}

} //return longest\_word; //i = 1, i + (k-1)

}

return string;

}

Problem 4:

Welcome. In this kata, you are asked to square every digit of a number.

For example, if we run 9119 through the function, 811181 will come out, because 92 is 81 and 12 is 1.

Ans:

function squareDigits(num){

//may the code be with you

var strNum = String(num);

var result='';

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

result= result+strNum[i]\*strNum[i];

}

return Number(result);

}

Problem 5:

Write a method that takes an array of consecutive (increasing) letters as input and that returns the missing letter in the array.

You will always get an valid array. And it will be always exactly one letter be missing. The length of the array will always be at least 2.

The array will always contain letters in only one case.

Ans:

function findMissingLetter(array)

{

var arr=array.map(function(char){

return char.toLowerCase();

});

var alphabet = 'abcdefghijklmnopqrstuvwxyza';

var arrNum = arr.map(function(char) {

return alphabet.indexOf(char);

});

var missingNum = null;

for (var i = 0; i < arrNum.length - 1; i++) {

if (arrNum[i] !== arrNum[i + 1] - 1) {

missingNum = arrNum[i] + 1;

}

}

var missingAlpha = alphabet[missingNum];

if (array[0] === array[0].toLowerCase()) {

return missingAlpha;

} else {

return missingAlpha.toUpperCase();

}

}

Problem 6:

Complete the function scramble(str1, str2) that returns true if a portion of str1 characters can be rearranged to match str2, otherwise returns false.

Ans:

function StringScramble(str1, str2) {

var arr = str2.split('');

var res = "";

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

var index = str1.indexOf(arr[i]);

res = index;

if (index < 0) {

return false;

}

str1 = str1.substring(0, index) + str1.substring(index + 1);

}

return true;

}