翻转字符串

function reverseString(str) {

// 请把你的代码写在这里

return str.split('').reverse().join('');

}

reverseString("hello");

String 全局对象是一个用于字符串或一个字符序列的构造函数。

split() 方法使用指定的分隔符字符串将一个String对象分割成字符串数组，以将字符串分隔为子字符串，以确定每个拆分的位置。

str.split([separator[, limit]])

separator

指定表示每个拆分应发生的点的字符串。separator 可以是一个字符串或正则表达式。

如果纯文本分隔符包含多个字符，则必须找到整个字符串来表示分割点。

如果在str中省略或不出现分隔符，则返回的数组包含一个由整个字符串组成的元素。

如果分隔符为空字符串，则将str原字符串中每个字符的数组形式返回。

limit

一个整数，限定返回的分割片段数量。

当提供此参数时，split 方法会在指定分隔符的每次出现时分割该字符串，但在限制条目已放入数组时停止。

如果在达到指定限制之前达到字符串的末尾，它可能仍然包含少于限制的条目。

新数组中不返回剩下的文本。

reverse() 方法将数组中元素的位置颠倒,并返回该数组。该方法会改变原数组。

join() 方法将一个数组（或一个类数组对象）的所有元素连接成一个字符串并返回这个字符串。

如果数组只有一个项目，那么将返回该项目而不使用分隔符。

计算一个整数的阶乘

function factorialize(num) {

// 请把你的代码写在这里

if (num>0)

{return (num \* factorialize(num - 1));}

else

return (1);

}

factorialize(5);

检查回文字符串

function palindrome(str) {

// 请把你的代码写在这 里

var newStr = str.replace(/[\ |\~|\`|\!|\@|\#|\$|\%|\^|\&|\\*|\(|\)|\-|\\_|\+|\=|\||\\|\[|\]|\{|\}|\;|\:|\"|\'|\,|\<|\.|\>|\/|\?]/g,"");

var newNewStr = newStr.toLowerCase();

var spl = newNewStr.split("");

var newSpl = spl.reverse();

var newRel = newSpl.join("");

if (newNewStr == newRel) {

return true;

} else return false;

}

palindrome("eye");

replace() 方法返回一个由替换值（replacement）替换一些或所有匹配的模式（pattern）后的新字符串。模式可以是一个字符串或者一个正则表达式，替换值可以是一个字符串或者一个每次匹配都要调用的函数。

原字符串不会改变。

toLowerCase() 会将调用该方法的字符串值转为小写形式，并返回。

找出最长单词

function findLongestWord(str) {

var arr=str.split(/\s+/);

var arrNum=[];

for(var i=0;i<arr.length;i++){//遍历数组

arrNum.push(arr[i].length);//查找数组长度

}

return Math.max.apply(null,arrNum);//取出最长的数组

}

findLongestWord("The quick brown fox jumped over the lazy dog");

split() 方法使用指定的分隔符字符串将一个String对象分割成字符串数组，以将字符串分隔为子字符串，以确定每个拆分的位置

句中单词首字母大写

function titleCase(str) {

//将字符串转化为数组并将其小写化

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

//遍历数组

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

//把str的第i个char值赋值给char，类似于数组的挨个查找。

var char = arr[i].charAt(0);

//将数组arr的第i个元素中的第一个符合变量char的字串变为大写

arr[i] = arr[i].replace(char, function replace(char){

return char.toUpperCase();

});

}

return arr.join(' ');

//return str;

}

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

split() 方法使用指定的分隔符字符串将一个String对象分割成字符串数组，以将字符串分隔为子字符串，以确定每个拆分的位置。

找出多个数组中的最大数

function largestOfFour(arr) {

// 请把你的代码写在这里

//创建数组容器

var arr1=[];

//二维数组长度遍历

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

var a= arr[i][0];

for(var j=1;j<arr[i].length;j++){

//arr[i][j]?a:arr[i][j]是一个整体

//arr[i][j]是否等于a 等于就是arr[i][j].

//前面判断a是否大于arr[i][j];

a= a>arr[i][j]?a:arr[i][j];

}

arr1[i]=a;

}

return arr1 ;

}

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

检查字符串结尾

function confirmEnding(str, target) {

//判断首字母是否是大写

//是则返回true 不是返回false

var a=target.length;

var s =str.substr(-a);

return s==target;

}

confirmEnding("Bastian", "n");

substr() 方法返回一个字符串中从指定位置开始到指定字符数的字符。

重复输出字符串

function repeat(str, num) {

var newStr="";

if(num<0){

newStr = "";

}else{

for(var i=0;i<num;i++){

newStr += str;

}

}

return newStr;

}

// 请把你的代码写在这里

repeat("abc", 3);

截断字符串

function truncate(str, num) {

// 请把你的代码写在这里

var len = str.length;//建立一个长度

if(num >= len){//判断num的长度是否大于等于我所建立的长度；

return str; //如果符合条件则返回str 第一个参数

}

else if(num-3 < len && num-3 > 0){ //如果不符合但是符合num参数-3小于len 或者 num-3大于0的话

return str.substr(0,num-3)+"..."; //返回len+... 把多余的部分用...表示 str.substr(start[, length])

}

I

I I0I)+"...";//否则返回num的长度和....

}

}

I"A-I I A green and yellow basket", 11);

slice() 方法提取一个字符串的一部分，并返回一新的字符串。

IbeginSliceI[, endSlice])

猴子吃香蕉, 分割数组

function chunk(arr, size) {

var newarr = []; //建立新的空数组

for (var i = 0; i < arr.length; i+=size){//找到arr参数的数组长度

newarr.push(arr.slice(i, i+size));//添加 push() 方法将一个或多个元素添加到数组的末尾，并返回该数组的新长度。slice() 方法返回一个新的数组对象，这一对象是一个由 begin和 end（不包括end）决定的原数组的浅拷贝。原始数组不会被改变。

}

console.log(newarr);

return newarr;

//问题解释 2是size 进行分割

//最后的数字就是多少个一组

}

chunk(["a", "b", "c", "d"], 2);

push() 方法将一个或多个元素添加到数组的末尾，并返回该数组的新长度。

slice() 方法返回一个新的数组对象，这一对象是一个由 begin和 end（不包括end）决定的原数组的浅拷贝。原始数组不会被改变。

比较字符串

（蛤蟆可以吃队友，也可以吃对手）

function mutation(arr) {

// 请把你的代码写在这里

var str=arr[1].toLowerCase().split("");//第二个字符串转换为小写

for(var i=0;i<str.length;i++){//遍历数组

if(arr[0].toLowerCase().indexOf(str[i])<0){//第一个字符串转换为小写 并且索引字符串 判断有没有完全相同

//负一就是不相同的

//indexOf() 方法返回调用 String 对象中第一次出现的指定值的索引，开始在 fromIndex进行搜索。

//如果未找到该值，则返回-1。

return false;//上面判断有没有相同的 如果没有完全相同 则返回false

}

}

return true;//如果完全相同 返回true

}

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

indexOf() 方法返回调用 String 对象中第一次出现的指定值的索引，开始在 fromIndex进行搜索

过滤数组假值

（真假美猴王）

function bouncer(arr) {

arr = arr.filter(Boolean);

return arr;

}

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

Boolean对象是一个布尔值的对象包装器。