1 判断字符串是否回文。

考察js字符串几个方法。

**function** checkPalindrom(str) {  
 **return** str == str.split('').reverse().join('');  
}

其中 str.split(‘’),通过空格分割，分成字符串数组。

str.reverse(),颠倒数组中数据，返回新数组。

Join方法，把数组中所有元素放入一个字符串。

2．数组去重

**let** unique = **function** (arr) {  
 **let** hashTable = {};  
    **let** data = [];  
    **for** (**let** i = 0, l = arr.length; i < l; i++) {  
 **if** (!hashTable[arr[i]]) {  
 hashTable[arr[i]] = **true**;  
            data.push(arr[i]);  
        }  
 }  
 **return** data  
}  
console.log(unique([3, 4, 3, 2, 4, 3]));

data为无重新数组，hashTable为对象。若对象中没有这个arr[i],就在

这个对象中添加这个数，然后把这个数push到新数组data，打印data。

也可以不用hashtabel直接，

Arr[i]push进data[]，每添加一次，在data里遍历一遍如果无这个数就push

进去，不然则不添加。

3检测字符串中出现最多的字符。

**function** findMaxDuplicateChar(str) {  
 **if** (str.length == 1) {  
 **return** str;  
    }  
 **let** charObj = {};  
    **for** (**let** i = 0; i < str.length; i++) {  
 **if** (!charObj[str.charAt(i)]) {  
 charObj[str.charAt(i)] = 1;  
        } **else** {  
 charObj[str.charAt(i)] += 1;  
        }  
 }  
 **let** maxChar = '',  
            maxValue = 1;  
    **for** (**var** k **in** charObj) {  
 **if** (charObj[k] >= maxValue) {  
 maxChar = k;  
            maxValue = charObj[k];  
        }  
 }  
 **return** maxChar;  
}  
console.log(findMaxDuplicateChar('afjghdfraaaasdenas'))

字符串 charAt方法。

返回字符串任意位置字符，如 a=’asdfg’, a.charAt(0)=’a’ a.charAt(1)=’s’

然后也是建立一个存放字符的对象charobj，如果立面已经有这个字符属性，则字符属性值

+1，若无，置一。

然后用for …in遍历对象charobj。

补充for… in方法语句用于对数组或者对象的属性进行循环操作。

，定义最大值变量maxValue为1，若里面有属性的值大于maxvalue，则maxvalue为

那个属性的值，同时对maxChar置为那为那个属性。

然后返回maxchar即为找出最大的出现次数最多的字符，。

同理也可以用数组代替对象来实现。

4，冒泡排序

**function** bubbleSort(arr) {  
 **for** (**var** i = 0, l = arr.length; i < l - 1; i++) {  
 **for** (**var** j = i + 1; j < l; j++) {  
 **if** (arr[i] > arr[j]) {  
 **var** tem = arr[j];  
                arr[j] = arr[i];  
                arr[i] = tem;  
            }  
 }  
 }  
 **return** arr;  
}  
console.log(bubbleSort([3, 4, 5, 2, 3, 4, 4, 6766, 6]));

大数沉底，外层到length-1结束。

快速排序

**function** quickSort(arr) {  
 **if** (arr.length <= 1) {  
 **return** arr;  
    }  
 **let** leftArr = [], rightArr = [], q = arr[0];  
    **for** (**var** i = 1, l = arr.length; i < l; i++) {  
 **if** (arr[i] > q) {  
 rightArr.push(arr[i]);  
        } **else** {  
 leftArr.push(arr[i]);  
        }  
 }  
 **return** [].concat(quickSort(leftArr), [q], quickSort(rightArr));  
}  
console.log(quickSort([3, 4, 5, 2, 3, 4, 4, 6766, 6]));