# 函数返回值：

1.function a(){

alert(1256);

return 1;

}

var b=a();//弹出1256

alert(a);//弹出整个函数

var b=a;//无弹出

alert(a());//先弹出1256， 在弹出1

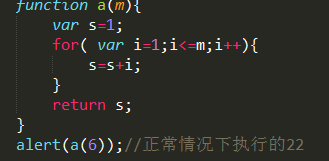
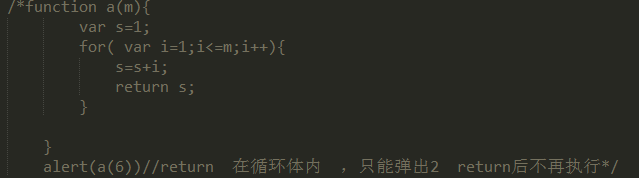
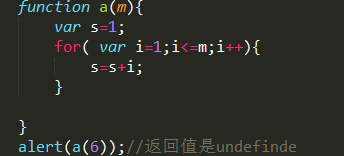
2.function a(){

return 2;

alert(569);

}

alert(a());//只能弹出2 ，569弹不出 因为一旦return 后面的就不在执行了

1. 
2. 
3. 函数内没有返回（return）
4. 递归：程序调用自身的编程技巧称为递归（ recursion）

递归：求积

function a(m){

if(m<=1){

return 1;

}else{

return m\*a(m-1);//此处调用了函数自己

}

}

alert(a(6)); 6\*5\*4\*3\*2\*1

求和

function a(m){

var s=0;

for(var i=1;i<=m;i++){

s=s+i;

}

return s;

}

alert(a(6));

递归求和：

function b(n){

if(n<=1){

return 1;

}else{

return n+b(n-1);

}

}

alert(b(6));

1. arguments对象是所有（非箭头）函数中都可用的局部变量，arguments对象不是一个 [Array](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Array" \o "entries() 方法返回一个新的Array Iterator对象，该对象包含数组中每个索引的键/值对。) 但除了[长度](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Functions/arguments/length)之外没有任何Array属性。例如，它没有 [pop](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Global_Objects/Array/pop" \o "JavaScript/Reference/Global_Objects/Array/pop) （**pop()**方法从数组中**删除**最后一个**元素**，并**返回**该元素的**值**。此方法**更改**数组的**长度**。）方法。但是它可以被转换为一个真正的Array。

function sum(m,n){

alert(arguments.length);//输入的参数有几个，arguments 长度就是多少

}

sum(6,6)

function a(){

alert(arguments[1]);//显示2，

}

a(1,2,3,6,5,4,7,8,7)

任何数的求和

function a(){

var s=0;

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

s+=arguments[i];

}

return s;

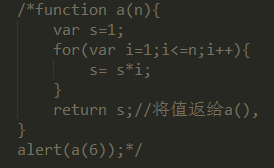
}

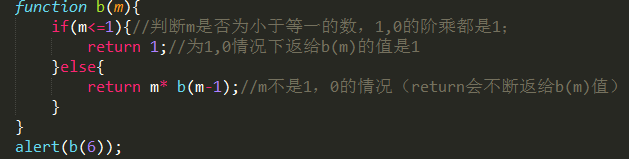
alert(a(1,6,9,8,7,5));

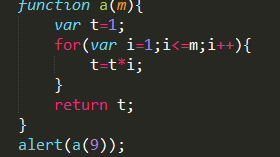
1. 提取设置属性

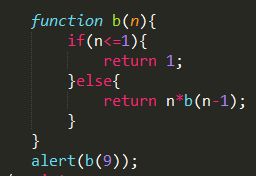


1. 获取行间样式
2. （递归）阶乘

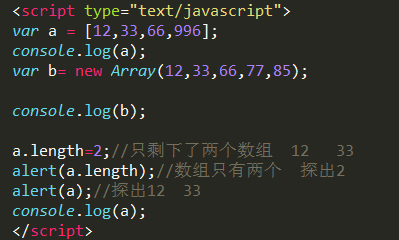








1. 改变数组长度



var a = [12,33,66,996]; 把a的length定义为2，就会只留下前两位

十二：元素组操控

1.合并数组：var arr =[12,33,11,25];

var a=[12,33,55,66];

console.log(arr.concat(a));//concat(要合并的数组)

输出结果：[12, 33, 11, 25, 12, 33, 55, 66]（新创建了一个数组，原数组arr不变）

1. 在数组最后加上一位

arr.push(20); 结果[12, 33, 11, 25, 20]（不会有上次合并的数组，这是在arr上操作）

1. 在数组最前面插入一个数

arr.unshift(10); [10, 12, 33, 11, 25, 20]（在原数组上操作）

1. 删除数组最后一位

arr.pop(); [10, 12, 33, 11, 25]

1. 删除数组第一位

arr.shift(); [12, 33, 11, 25]

1. arr.splice(0,2);//arr.splice(起始,删除个数);/ 从第一位 （0）开始删除2个， 剩下11 25 （在原数组arr上操作）
2. arr.splice(0,2,11,22)//删除两个后插入11 22

结果 11 12

1. arr.join('-') 把数组用“-”链接起来，变成字符串 结果：11-22
2. split（按照什么来分割字符串，显示字符长度（不写显示原本长度））

十三：数组排序：

Sort（）能够进行排序 按照首位值排序 数字时候 用比较函数

var a=['adas','dfsfs','bthet','wgwew'];

a.sort();//按照首字母排序

console.log(a);

var b =[1554,2545,56,955652,75555];

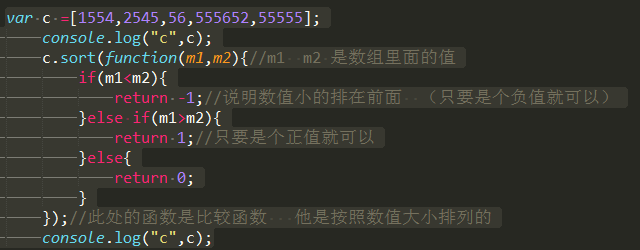
console.log(b);

b.sort();//按照首位数字大小排列，按字符处理

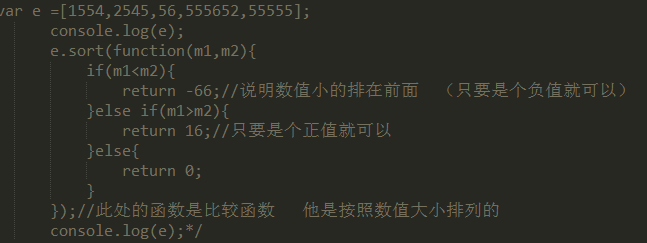
console.log(b);

结果：[1554, 2545, 56, 75555, 955652]

按照数值大小排序



[56, 1554, 2545, 55555, 555652]



M1<m2 return -66 (只要return一个负值就行）

M1>m2 return 16 (只要return一个争执就可以）

更简便的方法：

