目录

[1. 记忆函数-判断一个数是否是质数 1](#_Toc24406)

[2. 桶排序 把arr的值做为arr2的key 2](#_Toc25586)

[3. 快速排序 （一拆为二） 2](#_Toc29666)

[4. 插入排序 3](#_Toc31416)

[5. 希尔排序-分组加插入排序-长数组好性能 3](#_Toc21107)

[6. 选择排序 3](#_Toc5629)

[7. 冒泡排序 4](#_Toc31972)

*[8.](#_Toc16013)* [去掉数组重复的元素 4](#_Toc16013)

[9. 字母尾部加上单位或数字 6](#_Toc13736)

[10. 计算一个只有乘法和加法的表达式的值 6](#_Toc32389)

[11. 查找字符串中出现次数最多的字符，共出现几次? 6](#_Toc12818)

[12. 数组降维 7](#_Toc25326)

[13. 求出最小值和最小值索引 7](#_Toc19460)

[14. 无名 8](#_Toc17203)

[15. 闭包-判断函数的输出结果 9](#_Toc5637)

[16. 判断函数的输出结果 9](#_Toc17761)

[17. 阿里面试题-this指向 11](#_Toc3336)

[18. 美团面试题-全局变量和局部变量同名 11](#_Toc3409)

### 记忆函数-判断一个数是否是质数

**"use strict"**;

**let *isparime***=(**function**(){  
 **let** hash={};  
 **return function**(n){  
 **if**(!*isNaN*(n)&&n>1){  
 **if**(n<=3) **return true**;  
 **else if**(n%2==0) **return false**;  
 **else if**(hash[n]!==**undefined**){  
 ***console***.log(**"不用执行for循环。。"**);  
 **return** hash[n];  
 }**else**{  
 ***console***.log(**"执行for循环"**);  
 **for**(**var** i=3;i<=***Math***.sqrt(n);i+=2){  
 **if**(n%i==0){  
 hash[n]=**false**;  
 **return false**;  
 }  
 }  
 hash[n]=**true**;  
 **return true**;  
 }  
 }**else**{  
 **throw new *Error***(**"必须输入>1的数字"**);  
 }  
 }  
})();  
***console***.log(***isparime***(3));  
***console***.log(***isparime***(499));  
***console***.log(***isparime***(499));

### 桶排序 把arr的值做为arr2的key

**let *arr*** = [2,5,9,4,8,36,12]  
**let *arr2*** = [];  
**for** (**let *i*** = 0; ***i*** < ***arr***.**length**; ***i***++) {  
 **let *key*** = ***arr***[***i***];  
 ***arr2***[***key***] = 1;  
}  
**for** (**let *j* in *arr2***) {  
 ***console***.log(***j***);  
}  
***console***.log(***arr2***)

1. 快速排序 （一拆为二）  
   **function** *quickSort*(arr){  
    **if**(arr.**length** <= 1){  
    **return** arr;  
    }  
    **var** left = [];  
    **var** right = [];  
    **var** midIndex = *parseInt*(arr.**length** / 2);  
    **var** mid = arr[midIndex];  
    **for**(**var** i = 0 ; i < arr.**length** ; i++){  
    **if**(i == midIndex) **continue**;  
    **if**( arr[i] < mid){  
    left.push(arr[i])  
    }**else**{  
    right.push(arr[i]);  
    }  
    }  
    **return** *quickSort*(left).concat([mid],*quickSort*(right));  
   }  
   ***console***.log(*quickSort*([5,6,3,15,7,9]))

### 插入排序

**let *arr***=[4,2,5,3,1,8,9,12,4,77];  
**function** *insertStort*(arr) {  
 **for**(**let** i=1;i<arr.**length**;i++){  
 **let** t=arr[i];  
 **let** p=i-1;  
 **while**(p>=0&&arr[p]>t){  
 arr[p+1]=arr[p];  
 p--;  
 }  
 arr[p+1]=t  
 }  
}  
*insertStort*(***arr***);  
***console***.log(***arr***)

### 希尔排序-分组加插入排序-长数组好性能

**let *arr*** = [7,4,5,3,1,0,9,2,6,10,11,12,15,18,19,20,14,17,13]  
**let *len*** = ***arr***.**length  
for** (**let *gap*** = ***Math***.floor(***len*** / 2); ***gap*** > 0; ***gap*** = ***Math***.floor(***gap*** / 2)) {  
 **for**(**let *i*** = ***gap***; ***i***< ***len***; ***i***++) {  
 **let *cur*** = ***arr***[***i***]  
 **let *j*** = ***i*** - ***gap* while**(***j*** >= 0 && ***arr***[***j***] > ***cur***) {  
 ***arr***[***j*** + ***gap***] = ***arr***[***j***]  
 ***j*** -= ***gap*** }  
 ***arr***[***j*** + ***gap***] = ***cur*** }  
}  
***console***.log(***arr***)

### 选择排序

**let *arr*** = [9, 6, 1, 23, 5, 8, 11, 78, 45];  
**for** (**let *i*** = 0; ***i*** < ***arr***.**length** - 1; ***i***++) {  
 **let *temp* for** (**let *j*** = ***i*** + 1; ***j*** < ***arr***.**length**; ***j***++) {  
 **if** (***arr***[***i***] > ***arr***[***j***]) {  
 ***temp*** = ***arr***[***i***];  
 ***arr***[***i***] = ***arr***[***j***];  
 ***arr***[***j***] = ***temp***;  
 }  
 }  
}  
***console***.log(***arr***)

### 冒泡排序

**let *arr*** = [8,1,2,5,0,5,7]  
**for**(**let *i*** = 0; ***i*** < ***arr***.**length**; ***i***++) {  
 **for**(**let *j*** = 0; ***j*** < ***arr***.**length** - ***i***; ***j***++) {  
 **let *temp* if**(***arr***[***j***] > ***arr***[***j*** + 1]) {  
 ***temp*** = ***arr***[***j*** + 1]  
 ***arr***[***j*** + 1] = ***arr***[***j***]  
 ***arr***[***j***] = ***temp*** }  
 }  
}  
***console***.log(***arr***)

1. 去掉数组重复的元素**let *arr***=[3,1,3,1,4,2,3,9,4,5,2,8,2];  
    **let *r***=[];  
    **let *hash***={};  
    **for**(**let *i***=0;***i***<***arr***.length;***i***++){  
    **if**(***hash***[***arr***[***i***]]==**undefined**){  
    ***hash***[***arr***[***i***]]=1;  
    ***r***.push(***arr***[***i***])  
    }  
    }  
    ***console***.log(***r***)

**ES6去重：**

***// 利用Map对象和数组的filter方法*** *// 1.Map对象是ES6提供的一个新的数据结构，其中has的办法是返回一个布尔值，表示某个值是否存在当前的Mp对象之中，set的办法是给Map对象设置key/value。  
 // 2.filter() 方法创建一个新的数组，新数组中的元素是通过检查指定数组中符合条件的所有元素。  
 // 所以说，Map对象结合filter方法可以达到数组去重的效果~***let *arr*** = [2,2,2,**null**,**null**]  
**const *map*** = **new *Map***()  
**let *newArr*** = ***arr***.filter(item => !***map***.has(item) && ***map***.set(item, 1))  
***console***.log(***newArr***)

***// 利用from和 set去重*** *// 1.Set是ES6新提供的数据结构，类似于数组，但是本身没有重复值。  
// 2.Array.from方法用于将两类对象转为真正的数组：类似数组的对象（array-like object）和可遍历（iterable）的对象（包括 ES6 新增的数据结构 Set 和 Map）。  
// 所以set结合Array.from同样可以达到数组去重的效果。不过需要注意的是，主流浏览器像Chrome,Firfox,Opera，Safari，包括微软的Edge,都是支持的，但是唯独IE系列不支持。***let *arr*** = [1,2,2,1,**undefined**,**undefined**]  
**let *newArr*** = ***Array***.from(**new *Set***(***arr***))  
***console***.log(***newArr***)

***// 利用set + 扩展运算符...*****let *arr*** = [1, 1,3,3,3,***NaN***,***NaN***]  
**let *newArr*** = [...new ***Set***(***arr***)]  
***console***.log(***newArr***)

### 字母尾部加上单位或数字

**let *str***=**"ryan is not a good man"**;  
*//ryan5 is6 not7 a8 good9 man10***let *n***=5;  
***str***=***str***.replace(/\b[a-z]+\b/ig,**function**(kw){  
 **return** kw+***n***++;  
});  
***console***.log(***str***);

### 计算一个只有乘法和加法的表达式的值

**function** *calc*(exp){  
 **let** arr=exp.split(**"+"**);  
 **let** sum=0;  
 **for**(**let** i=0;i<arr.length;i++){  
 **if**(arr[i].indexOf(**"\*"**)==-1)  
 sum+=*parseFloat*(arr[i]);  
 **else**{  
 **let** r=1;  
 **let** arr2=arr[i].split(**"\*"**);  
 **for**(**let** j=0;j<arr2.length;j++){  
 r\*=arr2[j]  
 }  
 sum+=r;  
 }  
 }  
 **return** sum;  
}  
***console***.log(*calc*(**"1+23\*2+3\*3\*3"**));

**另一种简单的方法：**

***console***.log(*eval*(**"1+23\*2+3\*3\*3"**)))

### 查找字符串中出现次数最多的字符，共出现几次?

**let *str***=**"abdcbcdadcabce"**;  
**let *hash***={};  
**for**(**let *i***=0;***i***<***str***.length;***i***++){  
 **if**(***hash***[***str***[***i***]]!==**undefined**)  
 ***hash***[***str***[***i***]]+=1;  
 **else  
 *hash***[***str***[***i***]]=1;  
}  
**let *zi***=**""**,***count***=0;  
**for**(**let *key*** in ***hash***){  
 **if**(***hash***[***key***]>***count***){  
 ***zi***=***key***;  
 ***count***=***hash***[***key***];  
 }  
}  
***console***.log(**"出现最多的字符:"**+***zi***+**",共"**+***count***+**"个"**);

### 数组降维

**let *arr***=[  
 [0,0,0,0],  
 [0,0,0,0],  
 [0,0,0,0],  
 [0,0,0,0]  
];  
*//arr=String(arr).split(",");****arr***=***Array***.prototype.concat.apply([],***arr***);  
***console***.log(***arr***);

***// 使用 reduce、concat 和递归无限反嵌套多层嵌套的数组*****var *arr1*** = [1,2,3,[1,2,3,4, [2,3,4]]];  
  
**function** *flattenDeep*(arr1) {  
 **return** arr1.reduce((acc, val) => ***Array***.isArray(val) ? acc.concat(*flattenDeep*(val)) : acc.concat(val), []);  
}  
*flattenDeep*(***arr1***);  
*// [1, 2, 3, 1, 2, 3, 4, 2, 3, 4]*

### 求出最小值和最小值索引

*// 求最小值***let *arr*** = [10, 2,3,5,9,8,7]  
*// let min = Math.min(...arr)  
// console.log(min)***let *min*** = ***arr***[0]  
***arr***.forEach(item => {  
 **if**(item < ***min***) ***min*** = item  
})  
***console***.log(***min***)  
*// 求出最小值索引***let *i*** = ***arr***.findIndex(item => item === ***min***)  
***console***.log(***i***)

### 无名

*//let f=function(){ let a= b=1; }  
//f();  
//console.log(b);//1  
//console.log(a);//报错  
  
//let f=function(){ let a=b=1; }  
//setTimeout(f,0);  
//console.log(b);//报错  
////f();  
  
//let a,b=0,fn=function(){let a= b=2};  
//fn();  
//console.log(a);//undefined  
//console.log(b);//2***let *emp***={  
 work:**function**(){  
 **let** sum=0;  
 **for**(**let** i=0;  
 i<***arguments***.length&&***arguments***[0]>0;  
 i++  
 ){  
 sum+=***arguments***[i]  
 +***arguments***.callee(--***arguments***[i])  
 }  
 **return** sum;  
 }  
};  
***console***.log(***emp***.work(3,2,1));

### 闭包-判断函数的输出结果

**let *a***=0,***b***=0;  
**function** *A*(a){*//外层函数  
 //外层函数向外返回内层函数——给全局变量A赋值  
 A*=**function**(b){***console***.log(a+b++);}*//内层函数* ***console***.log(a);  
}  
*A*(1);*//1  
//A:function(b){console.log(a+b++);}(a=1)  
A*(12);*//13?*

### 判断函数的输出结果

**1、**

**function** *fun*(n,o){  
***console***.log(o);*//输出第2个参数***return** {  
fun:**function**(m){*//(保护第1个参数)***return** *fun*(m,n);  
}  
}  
}  
**let *b***=*fun*(0)*//undefined  
/\*b:{fun(){return fun(m,n=0)}}\*/*.fun(1)*//0  
/\*b:{fun(){return fun(m,n=1)}}\*/*.fun(2)*//1  
/\*b:{fun(){return fun(m,n=2)}}\*/*.fun(3);*//2***let *c***=*fun*(0)*//undefined*.fun(1);*//0  
/\*c:{fun(){return fun(m,n=1)}}\*/****c***.fun(2);*//1****c***.fun(3);*//1***let *a***=*fun*(0);*//undefined  
/\*a:{fun(){return fun(m,n=0)}}\*/****a***.fun(1);*//0****a***.fun(2);*//0****a***.fun(3);*//0*

2、

**let *arr***=[];  
**function** *fun*(){  
 **for**(**var** i=0;i<4;i++){  
 **let** x={};  
 *//x.no=i;  
 //x.text=arr[i];* x.fun=**function**(){***console***.log(i)};  
 ***arr***.push(x);  
 }*//i=4  
 /\*  
 arr:[  
 {fun(){console.log(i)}},  
 {fun(){console.log(i)}},  
 {fun(){console.log(i)}},  
 {fun(){console.log(i)}},  
 ]  
 \*/*}  
*fun*();  
***arr***[0].fun();*//4****arr***[1].fun();*//4****arr***[2].fun();*//4****arr***[3].fun();*//4*

3、

**let *funs***=(**function**(){  
 **for**(var i=0,arr=[];i<3;i++){  
 arr[i]=**function**(){***console***.log(i)};  
 }*//i=3* **return *arr***;  
})();  
*//funs:[  
// 0: function(){console.log(i)},  
// 1: function(){console.log(i)},  
// 2: function(){console.log(i)}  
// ]****funs***[0]();*//3****funs***[1]();*//3****funs***[2]();*//3*

### 阿里面试题-this指向

**var *length*** = 100  
**function** *f1*() {  
 ***console***.log(**this**.**length**)  
}  
**var *obj*** = {  
 **length**: 10,  
 f2: **function** (f1) {  
 f1() *// 没有点，无调用者 默认指向window 则length为100* arguments[0]() *// 有调用者arguments, this指向arguments 其length等于2* }  
}  
***obj***.f2(*f1*, 1)

### 美团面试题-全局变量和局部变量同名

**var *a*** = 10  
**function** *f1*() {  
 **var** b = 2 \* a  
 **var** a = 20  
 **var** c = a + 1  
 ***console***.log(b) *// a === undefined 所以输出NAN* ***console***.log(c) *// a = 20 输出21*}  
*f1*()