Js专题

1. 冒泡排序？

function bubbleSort(data){

var temp=0;

for(var i=data.length;i>0;i--){

for(var j=0;j<i-1;j++){

if(data[j]>data[j+1])

{

temp=data[j];

data[j]=data[j+1];

data[j+1]=temp;

}

}

}

return data;

}

1. 选择排序?

function selectionSort(data){

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

var min=data[i];

var temp;

var index=1;

for(var j=i+1;j<data.length;j++){

if(data[j]<min)

{

temp=data[j];

data[j]=min;

min=temp;

}

}

temp=data[i];

data[i]=min;

data[index]=temp

}

1. 快速排序?

function quickSort(arr){

if(arr.length==0)

return [];

var left=[];

var right=[];

var pivot=arr[0];

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

if(arr[i]<pivot){

left.push(arr[i]);

}

else{

right.push(arr[i]);

}

}

return quickSort(left).concat(pivot,quickSort(right));

}

1. 插入排序?

function insertSort(data){

var len=data.length;

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

var key=data[i];

var j=i-1;

while(j>=0&&data[j]>key){

data[j+1]=data[i];

j--;

}

data[j+1]=key;

}

return data;

}

1. 数组去重es5方法1?

var arr = [1,2,3,4,4,2,2,6,9,1,0];

var newArr = [];

var onOff = true;

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

onOff = true;

for(var j = 0;j<newArr.length;j++){

if(newArr[j]==arr[i]){

onOff = false;

}

}

if(onOff){

newArr.push(arr[i]);

}

}

console.log(newArr)

1. 数组去重es5方法2？

var arr = [1,2,3,4,4,2,2,6,9,1,0];

var obj = {};

arr.forEach((e,i)=>{ //把arr的信息统计到obj里面

if(obj[e]===undefined){ //其实是利用对象属性是否存在做了一层遍历

obj[e] = 1

}else{

obj[e]++

}

});

console.log(Object.keys(obj))

1. Es6数组去重？

利用ES6中Set不包含重复元素的思想，为数组创建set对象，再将set对象转换为数组。

8、如何编写高性能的 Javascript ？

1.使用位运算代替一些简单的四则运算。

2.避免使用过深的嵌套循环。

3.不要使用未定义的变量。

4.当需要多次访问数组长度时，可以用变量保存起来，避免每次都会去进行属性查找。

5.建议将对象进行缓存处理，特别是DOM访问是比较消耗资源的

9、如何判断一个对象是否属于某个类？

第一种方式是使用 instanceof 运算符来判断构造函数的 prototype 属性是否出现在对象的原型链中的任何位置。

第二种方式可以通过对象的 constructor 属性来判断，对象的 constructor 属性指向该对象的构造函数，但是这种方式不是很安全，因为 constructor 属性可以被改写。

第三种方式，如果需要判断的是某个内置的引用类型的话，可以使用 Object.prototype.toString() 方法来打印对象的

[[Class]] 属性来进行判断。

1. 对于 JSON 的了解？

JSON 是一种基于文本的轻量级的数据交换格式。它可以被任何的编程语言读取和作为数据格式来传递。

在项目开发中，我们使用 JSON 作为前后端数据交换的方式。在前端我们通过将一个符合 JSON 格式的数据结构序列化为 JSON 字符串，然后将它传递到后端，后端通过 JSON 格式的字符串解析后生成对应的数据结构，以此来实现前后端数据的一个传递。

因为 JSON 的语法是基于 js 的，因此很容易将 JSON 和 js 中的对象弄混，但是我们应该注意的是 JSON 和 js 中的对象不是一回事，JSON 中对象格式更加严格，比如说在 JSON 中属性值不能为函数，不能出现 NaN 这样的属性值等，因此大多数的 js 对象是不符合 JSON 对象的格式的。

在 js 中提供了两个函数来实现 js 数据结构和 JSON 格式的转换处理，一个是 JSON.stringify 函数，通过传入一个符合 JSON 格式的数据结构，将其转换为一个 JSON 字符串。如果传入的数据结构不符合 JSON 格式，那么在序列化的时候会对这些值进行对应的特殊处理，使其符合规范。在前端向后端发送数据时，我们可以调用这个函数将数据对象转化为 JSON 格式的字符串。

另一个函数 JSON.parse() 函数，这个函数用来将 JSON 格式的字符串转换为一个 js 数据结构，如果传入的字符串不是标准的 JSON 格式的字符串的话，将会抛出错误。当我们从后端接收到 JSON 格式的字符串时，我们可以通过这个方法来将其解析为一个 js 数据结构，以此来进行数据的访问。

1. JavaScript 中的作用域与变量声明提升？

变量提升的表现是，无论我们在函数中何处位置声明的变量，好像都被提升到了函数的首部，我们可以在变量声明前访问到而不会报错。

造成变量声明提升的本质原因是 js 引擎在代码执行前有一个解析的过程，创建了执行上下文，初始化了一些代码执行时需要用到的对象。当我们访问一个变量时，我们会到当前执行上下文中的作用域链中去查找，而作用域链的首端指向的是当前执行上下文的变量对象，这个变量对象是执行上下文的一个属性，它包含了函数的形参、所有的函数和变量声明，这个对象的是在代码解析的时候创建的。这就是会出现变量声明提升的根本原因。

1. 深浅拷贝？

浅拷贝指的是将一个对象的属性值复制到另一个对象，如果有的属性的值为引用类型的话，那么会将这个引用的地址复制给对象，因此两个对象会有同一个引用类型的引用。浅拷贝可以使用 Object.assign 和展开运算符来实现。

深拷贝相对浅拷贝而言，如果遇到属性值为引用类型的时候，它新建一个引用类型并将对应的值复制给它，因此对象获得的一个新的引用类型而不是一个原有类型的引用。深拷贝对于一些对象可以使用 JSON 的两个函数来实现，但是由于 JSON 的对象格式比 js 的对象格式更加严格，所以如果属性值里边出现函数或者 Symbol 类型的值时，会转换失败。

1. js 中的异步机制？

第一种最常见的是使用回调函数的方式，使用回调函数的方式有一个缺点是，多个回调函数嵌套的时候会造成回调函数地狱，上下两层的回调函数间的代码耦合度太高，不利于代码的可维护。

第二种是 Promise 的方式，使用 Promise 的方式可以将嵌套的回调函数作为链式调用。但是使用这种方法，有时会造成多个 then 的链式调用，可能会造成代码的语义不够明确。

第三种是使用 generator 的方式，它可以在函数的执行过程中，将函数的执行权转移出去，在函数外部我们还可以将执行权转移回来。当我们遇到异步函数执行的时候，将函数执行权转移出去，当异步函数执行完毕的时候我们再将执行权给转移回来。因此我们在 generator 内部对于异步操作的方式，可以以同步的顺序来书写。使用这种方式我们需要考虑的问题是何时将函数的控制权转移回来，因此我们需要有一个自动执行 generator 的机制，比如说 co 模块等方式来实现 generator 的自动执行。

第四种是使用 async 函数的形式，async 函数是 generator 和 promise 实现的一个自动执行的语法糖，它内部自带执行器，当函数内部执行到一个 await 语句的时候，如果语句返回一个 promise 对象，那么函数将会等待 promise 对象的状态变为 resolve 后再继续向下执行。因此我们可以将异步逻辑，转化为同步的顺序来书写，并且这个函数可以自动执行。

1. js 获取原型的方法？

p.proto

p.constructor.prototype

Object.getPrototypeOf(p)

1. instanceof 的作用？

instanceof 运算符用于判断构造函数的 prototype 属性是否出现在对象的原型链中的任何位置。

16、大海中央的一座孤岛，上面本来没有老鼠。有一艘路过的船只在附近沉没了，船上 x 对 新出生的老鼠，顺着船只残骸漂流到小岛上。

1. 每两个月一次生产，每次出生 2对 老鼠

2. 老鼠的寿命是 5个月，也就是说一辈子可以生产 2次

3. 这个岛上，每个月因为神秘原因，假如存在 3个月大的老鼠，会死去一对

问 n个月后，岛上有多少只老鼠 (假设存活率 100% )

17、二叉搜索树的后序遍历？

输入一个整数数组，判断该数组是不是某二叉搜索树的后序遍历的结果。如果是则输出 Yes，否则输出 No。假设输入的数组的任意两个数字都互不相同。

对于一个合法而二叉树的后序遍历来说，最末尾的元素为根元素。该元素前面的元素可以划分为两个部分，一部分为该元素的左子树，

所有元素的值比根元素小，一部分为该元素的右子树，所有的元素的值比该根元素大。并且每一部分都是一个合法的后序序列，因此我们可以利用这些特点来递归判断。

1. 实现 IndexOf 方法？

function indexFun(array, val) {

if (!Array.isArray(array)) return;

let length = array.length;

for (let i = 0; i < length; i++) {

if (array[i] === val) {

return i;

}

}

return -1;

}

1. 请实现一个函数，将一个字符串中的空格替换成“%20”。例如，当字符串为 We Are Happy.则经过替换之后的字符串为 We%20Are%20Happy

使用正则表达式，结合字符串的 replace 方法将空格替换为 “%20”

str.replace(/\s/g,"%20")

20、字符串的排列？

输入一个字符串，按字典序打印出该字符串中字符的所有排列。例如输入字符串 abc，则打印出由字符 a,b,c 所能排列出来的所有

字符串 abc,acb,bac,bca,cab 和 cba。输入描述：输入一个字符串，长度不超过9（可能有字符重复），字符只包括大小写字母。？

思路：

我们可以把一个字符串看做是两个部分，第一部分为它的第一个字符，第二部分是它后面的所有字符。求整个字符串的一个全排列，可

以看做两步，第一步是求所有可能出现在第一个位置的字符，即把第一个字符和后面的所有字符交换。第二步就是求后面所有字符的一

个全排列。因此通过这种方式，我们可以以递归的思路来求出当前字符串的全排列。