1. js中的连等是从左往右依次连等
2. switch中括号中的要和case值全等(=== 值类型都要相等)，在switch的表达式和case的表达式是使用 === 严格相等运算符进行比较的
3. 在变量提升中if语句中的方法 要首先将其改写为 定义式 （var fun = function (){}）
4. 作用域及作用域链中函数的上级作用域只和函数的定义位置有关和函数在哪执行无关 并且自执行函数不会进行预解析
5. 在函数局部变量提升的过程中，如果没有传实参 且局部作用域也没有通过var来声明 那么就会从上级作用域去查找此变量。
6. 我们可以说变量对象包含了活动对象，活动对象就是作用域链上正在被执行和引用的变量对象。我们从活动对象的名称中也能看出 “执行、运行、激活” 等意味。你可以这样理解，整个代码的运行总有一个起始的对象吧，不管这个起始是变量还是函数，总要有一个称呼，虽然我们把执行环境中的变量和函数的总称叫做变量对象，但这不能反映代码的动态性，为了区别于普通的变量对象，我们创造了活动对象的概念。
7. For of 和for in for...of 语句创建一个循环来迭代可迭代的对象。在 ES6 中引入的 for...of 循环，以替代 for...in 和 forEach() ，并支持新的迭代协议。for...of 允许你遍历 Arrays（数组）, Strings（字符串）, Maps（映射）, Sets（集合）等可迭代的数据结构等，但是for of不能便利普通的对象。 For of 中的i值为每一项的属性值而forin中的i值为每一项的属性名(索引)
8. 在构造函数的原型中，要使用this来代替这个构造函数所实例化的对象，方便
9. 找对象中的key值和value值可以使用Object.keys()和Object.values()其返回值是一个数组
10. 在JavaScript语言中，()里面不能包含语句（只能是表达式），所以解析器在解析到function关键字的时候，会把它们当作function表达式，而不是正常的函数声明。
11. 立即执行函数不会进行预解析
12. Set的用法

定义：类似于数组的结构，且成员都是唯一的，且不能重复

声明：const set = new Set(arr)

属性：**constructor**：构造函数，返回Set size:返回成员的总数

方法：add()返回实例 delete()返回布尔值 has()返回布尔值 clear()清空所有成员 keys()返回以属性值为遍历器的对象 values()返回以属性值为遍历器的对象 entries()返回以属性值和属性值遍历的对象 forEach()遍历对象中的属性值 无返回值

1. 在定时器、事件监听、Ajax请求、跨窗口通信、Web Workers或者任何异步中，只要使用了回调函数，实际上就是在使用闭包。
2. 原生js的灵魂之问：<https://juejin.im/post/5dac5d82e51d45249850cd20>

 //查找一个数组所嵌套的层数

        function pires(arr,count=0) {            let ary=[];            let ary1=[]            for(let i = 0;i<arr.length;i++){

                   if(arr[i] instanceof Array){

                       ary1.push('true')

                    }

           }

           if(ary1.length==0){

               return count;

           }else{               let ary2 = arr[0];

                arr.shift();

             arr.push(...ary2)

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

                  if(arr[i] instanceof Array){

                      count++;

                      ary.push(arr[i]);                   }

        }

          }

           return pires(ary,count);

       }

1. JavaScript解释阶段便会确定作用域规则，因此作用域在函数定义时就已经确定了，而不是在函数调用时确定，但是执行上下文是函数执行之前创建的。执行上下文最明显的就是this的指向是执行时确定的。而作用域访问的变量是编写代码的结构确定的。
2. 驼峰命名法：(1)全部小写 单词与单词之间去下划线分隔

(2)大驼峰：每个单词首字母都大写 小驼峰：第一个单词的首字母小写，其余每个有意义的单词首字母大写

17.函数中的this与其在哪定义无关，和他在什么情况下调用有关。

18.生命周期：全局变量：除非被显示删除，不然他会一直存在

局部变量：在自声明起至函数执行完以后被销毁或者是被显示删除

19. 使用对象直接量方式定义的对象，它的构造器（constructor）

指向的是Object

 Person.prototype = {

            getName: function () {}

        }

20. 原型对象里有一个叫constructor的属性，它指向自己的构

造器（构造函数）所以，一般用constructor属性来获取当前对

象的构造函数