1. console.log(window.a)输出undefined，而console.log(a)，就报错，难道window.a是全局，a就不是全局？

答: 所有全局变量一定是保存在window中的

保存在window中的变量其实有两种访问方式:

可以以变量语法方式: 变量名

可以以window对象的成员形式使用: window.变量

比如: var a=10;

console.log(a) //10

因为所有没有前缀的，js都会自动加上window.前缀——仅限于全局作用域中执行的程序

比如: alert() -> 执行时被补全为window.alert()

再比如: parseInt() -> 执行时被补全为window.parseInt()

console.log(window.a) //10

如果没有声明a，执行console.log(a)和console.log(window.a)，一个报错，一个不报错？

语法规定:

如果直接访问一个不存在的变量，会报错！ReferenceError: a未定义

如果强行访问对象中一个不存在的属性，或访问数组中一个不存在的位置，不报错！而是返回undefined。

全局变量a，其实都有两个角色，一个是单纯的变量a，同时也是window对象中的成员。所以，以单独变量方式访问不存在的变量，则参考第一条规则，会报错。如果以window.a成员的方式访问变量a，永远不会报错！而是返回undefined。

补: var a和 window.a

1. 如果未声明变量a，则单独使用a会报错，而使用window.a不报错。

2. var a的变量不能用delete彻底删除

而用window.a强行添加的变量，才能被delete彻底删除

总结: 今后即使用全局变量，也要用window.变量。不要用var。因为如果不想用了，还可以用delete彻底删除

2. pay=null后，为什么变量pay还在？

答: 给pay变量赋值为null，只是清空变量的内容，而变量在window中不会被删除。

如果希望彻底删除变量: delete window.变量

但是: 删除不掉！

原因: pay是var pay创建的。如果想用delete彻底删除变量，创建变量时，必须用window.pay

3. console.log(a) //报错，为什么不是undefine?

a=10;

console.log(a)

答？因为a=10前，没有var。没有var，就不是声明语法，就不会会被声明提前。不会被声明提前，第一次使用时就没有a可用，就报错。a=10，只是一个赋值语句。赋值语句会强行创建变量a，但是，也只能在语句所在位置才创建a。因为声明提前规定，“”赋值留在原地“

4. 为什么加入overflow:hidden或添加元素设置为display:table，可防止高度坍塌？BFC？

答: Block Formatting Context

以块的形式，格式化显示元素的子内容

为什么: 启用BFC模式都是为了避免子元素被隐藏或遮挡。

如何启用BFC：

浮动元素：float 除 none 以外的值。

绝对定位元素：position (absolute、fixed)。

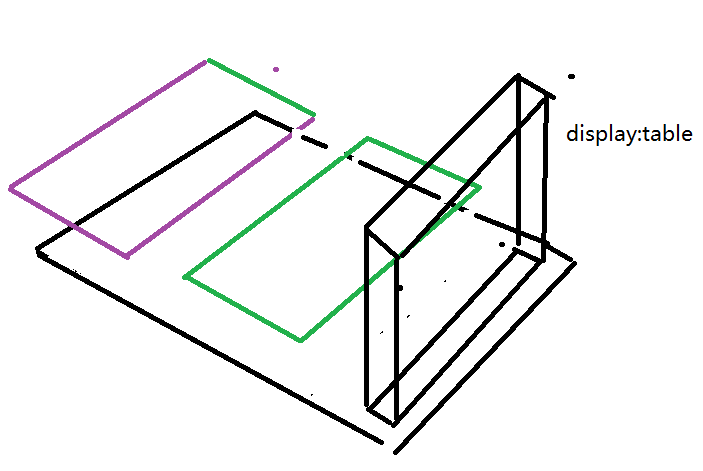
display 为 inline-block、table-cells、flex。

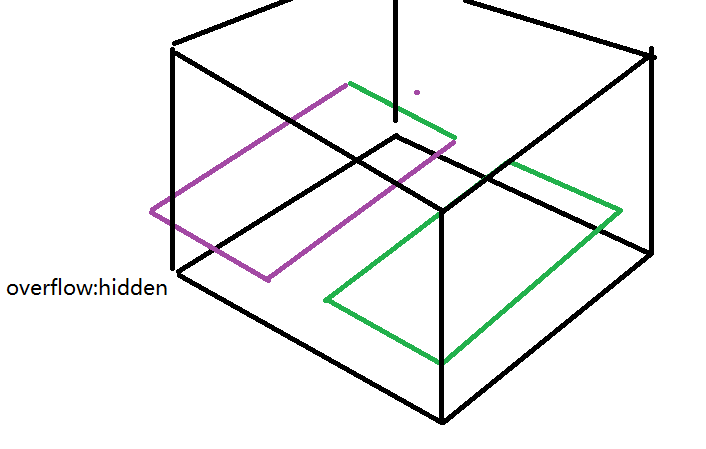
overflow 除了 visible 以外的值 (hidden、auto、scroll)

后果:

1. 父元素启用BFC，则范围必须包含所有子元素的范围。

2. 平级子元素启用BFC，则其他兄弟子元素不能覆盖在它之上。





5. 作用域到底是什么？

答: 作用域其实就保存变量的对象。只有两种作用域对象:

全局作用域对象: window  
 函数作用域对象：一般看不见，也没有。只有在调用函数时，在内存中临时创建。函数作用域对象中保存的是本次函数调用使用的局部变量。

6. 反复调用外层函数parent()，为什么变量值不变？和调用内层函数pay有什么差别？

答: 调用内层函数pay是一个小孩儿，反复花自己的红包

反复调用父母parent()，是反复生小孩儿，反复包红包。

7. 如何从外部访问到闭包中的变量:

答: 不让访问！如果能从外部随便访问，不叫闭包！

其实，是有办法看到的！

输出函数在内存中的结构： console.dir(pay)

dir可输出对象在内存中的详细存储结构

每个函数输出后都可看到一个属性[[scopes]]，这就是还有列表。里边在不调用时，暂时只有window或window和父母的作用域。

8. 为什么在修改之前输出对象的内容和修改后输出的内容都是修改之后的新内容？

答: console.log()和console.dir()不是给对象拍照的意思，不是留住对象瞬间的状态。而是输出对象的引用地址。

当程序结束后，我们点击三角展开对象时，控制台才临时去内存中找对应的对象的内容。此时，程序已经结束，所以无论点开哪个三角，看到的只能是最后的修改值。  
 解决: 分布注释，解开注释，调试程序，查看阶段的输出结果。不要最后一股脑都输出。

9. 为什么调用外层函数必须用变量接住返回值？

答: 如果想获得外层函数的返回结果，必须自己用变量接一下。如果不接，第一，无法获得内容。第二，不能重复使用。

因为只要重复使用的东西，都要保存在变量中

10. 如果外层函数只用一次，怎么办？

答: 绝大多数情况下，父母函数确实只要产生一次闭包，就不再使用。

解决: 如果一个函数只用一次，都可用匿名函数自调

比如: var pay=(function(){

var total=1000;

return function(money){

total-=money;

console.log(`本次花了${money}元，还剩${total}元`)

}

})();

pay(100);

结果: 父母函数调用一次后释放，但是孩子函数和父母函数的作用域对象都被保留下来。

11. getN(), nAdd(), getN() 为什么只有两个输出？

答: nAdd没有输出，也没有返回值。

12. 给nAdd前加上var，会怎样?

答: 报错，nAdd未定义。因为nAdd成了局部变量。函数外不可使用。

13. 在nAdd中改成console.log(n++)为什么输出999?

答: 因为是后++，先输出n旧的值，然后再对n+1

14. 禁止使用"name"作为变量名?

答：因为window中有一个固定的常量叫name，不可替代。

15. 为什么关联数组ym里明明有三个成绩，但是ym.length是0？

答：数组的length属性，紧紧记录数字下标的元素个数。自定义下标名称的元素，是不参与length统计的。

16. var lilei2=clone(lilei) 是什么意思？

答：调用clone函数，并将lilei对象作为参数，传入clone函数中处理。将clone函数的返回值，赋值给lilei2变量保存。

就好像，把玉米粒，放到爆米花机里，爆出爆米花，再将爆出的爆米花，装到袋子里保存起来。

17. nAdd=function(){ n++ }既然是全局变量，为什么可以访问fun(){}中的局部变量n呢？

答: nAdd=function(){ n++ }其实是2件事:

1.先创建一个函数——在fun内的——孩子是fun生的。

2. 然后才把fun生的孩子赋值给了全局变量，才从fun中出来的。

所以，因为function是fun生的，所以，使用自己父母的局部变量是情理之中的。

18. 既然对象不是作用域，那么对象中的属性，应该是全局变量呀？

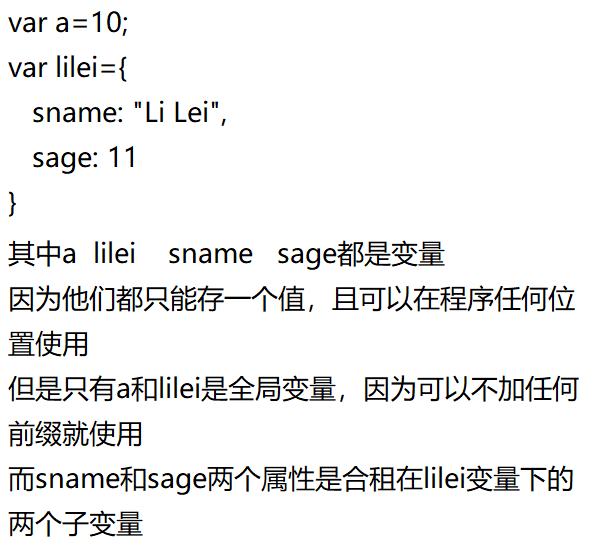
答: 对象中的属性，不是全局变量，是因为，直接使用属性名，不加对象名.前缀，是访问不到属性变量的

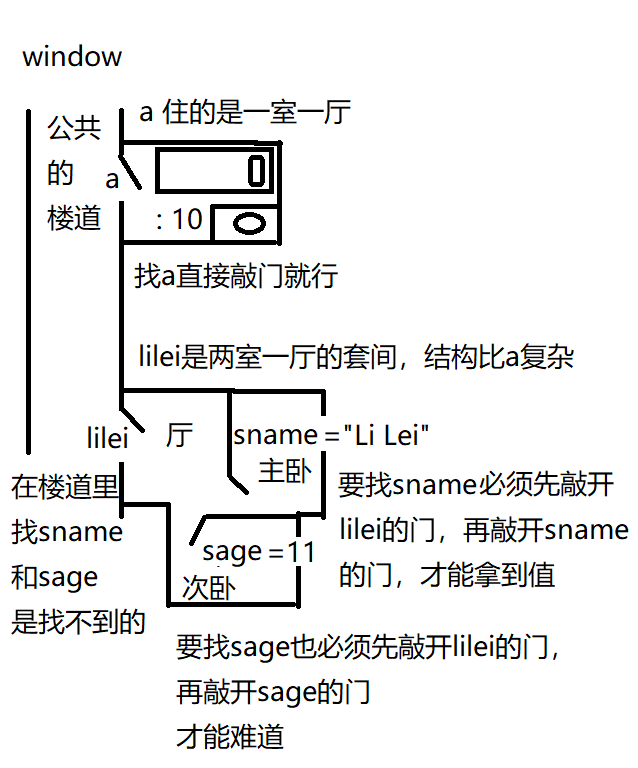
对象中的属性，也不是局部变量，因为，对象中的属性可以通过加对象名.前缀随时访问。而局部变量，只能在函数内访问，无论加什么，都不能在函数外访问。

那么对象中的属性，到底是什么变量？

答: 对象中的属性是一种套在对象变量中的子变量。可以当做变量一样使用，但是查找时，必须先找到对象，进入对象，才能找到属性变量。

就好像租房子： 有整租一室一厅，有合租两室一厅，合租三室一厅。





但是，sname和sage不是局部变量，更不是私有。因为，只要你可以敲开lilei的门，sname和sage你随便用，没有限制。

19: lilei中的方法intr()的好友列表中为什么只有两级作用域

答: 因为intr()方法外层的程序结构不是函数，只是一个普通的变量引用的一个普通对象，只不过结构比一般的只存一个值的变量结构复杂而已。所以不是一级作用域对象。不能作为好友列表中的一员。

函数的好友列表中，只能保存作用域对象，只要不是作用域的对象，都没有资格加入函数的好友列表中。

而闭包中，内层函数的外部，是一个外层函数，是一级作用域，既然是一级作用域对象，就有资格作为内层函数的好友列表中的一员。

20. var char=str[i]; arr[char]中的char为什么不加""？

答: char是不确定的，是从str中临时获取的变量值。只要是变量，就不应该加""。如果不加引号，程序在执行时，会动态将char变量的位置替换成char中保存的值。如果加了""，就无法动态改变变量值了。

21. arr["h"]===undefined，就可以判断数组中没有名为"h"的元素

答: 访问数组中任何一个位置的元素时，如果下标位置存在，就返回元素的值，如果下标位置不存在，也不会报错，而是返回undefined，所以经常用强行访问某个位置的元素，是否返回undefined来判断是否存在这个位置的元素。

22. 为什么===undefined，换成==有什么不同?

答: 因为==是自带隐式转换的等于比较，会将null也转为undefined。所以如果用==很容易误判null和undefined。而===是不带隐式转换的等于比较，所以可以区分null和undefined。因为null和undefined首先类型都不同。

23. 为什么数组中不存在arr["h"]，也能强行赋值？

答: js规定所有数组和对象，可在任何时候强行添加新属性/新成员，而不会报错！

24. for(var i=0;i<3;i++){} 循环结束后，i不应该是2吗？为什么是3？

答: 循环退出，一定是因为循环条件不满足。如果i还是2，那么2<3，循环条件还满足，怎么会退出循环呢？之所以循环退出了，一定是因为i<3条件不满足了。i只有=3时，循环条件才不满足了，循环才退出。

25. 为什么每个对象都有sname和sage属性，为什么不放在原型对象中公用呢？

答: 要公用的属性，必须连值都要完全相同，但是sname和sage是每个子对象各自不同的，所以不应该公用。

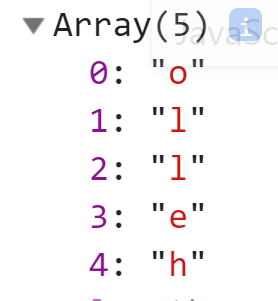
26. parent()()和var a=parent(); a()，两个输出结果是一样的，有什么差别？  
 答: parent()()，只能调用一次返回的内层函数，调用后，立刻释放。因为没有变量接住。

后边用a接住内层函数对象，就可反复使用。

27. var getN=fun(); fun()(); getN(); nAdd(); getN(); 结果是两个999，为什么？

答: nAdd是全局变量，第二次fun()时，又生成了一个nAdd把第一个fun()生成的nAdd覆盖了。又因为第二次fun()的nAdd和第一次fun()生的getN()不是一次生的，所以各自使用各自的红包，毫无关系。新的nAdd()影响不了旧的getN()的值。

28. 为什么arr.reverse()调用前输出的arr，却是已经翻转过的！



答: console.log(对象/数组)其实不是给对象拍照！而是输出对象的地址！

当程序执行完，我们去点地址旁边的时，才临时去内存中找到这个对象，显示对象的内容。

因为此时程序已经执行完了，且内存中只有一个arr数组。所以，无论何时点开三角，看到的对象内容都是一样的，且都是最终的结果！

解决: 如果非要看到数组中间瞬时的状态: 2个办法

1. 刷新页面，会在控制台，显示数组在中途执行时的照片。但是也不要展开。

2. 可以将数组或对象转为字符串，再输出

String(数组) vs toString(数组)

其实两个是极其相似的。

29. 函数为什么不能保存在变量中，而只能把函数地址地址保存在变量中？

答: 变量规定只能存1个值！只有简单的原始类型(number, string, boolean, null, undefined)值，才可直接保存在变量中。

而复杂的引用类型(数组，函数，对象等)，无法保存进变量中，只能在变量外开辟存储空间。将对象的地址保存回变量中。因为地址只是一个值，所以不违反变量的规定。

使用变量时，是通过变量中保存的地址找到远处的对象，来使用。

30. function add(){ ... } 和 Array.prototype.sum=function(){}都可以累加任意多个数的和，有什么区别？

答: function add(){ ... }，是所有程序都可随便使用的一个函数，且多个数值以零散的方式传入函数内求和。

Array.prototype.sum()，只允许数组家孩子使用，且只能遍历当前正在调用sum()函数的.前的数组中的元素，求和。无法随便传入sum()中参数值。

31. Array.prototype.sum=function(){}中的this.length，能不能换成arguments.length?

答: 不行。因为sum()函数的用法，是arr1.sum()，arr2.sum()，是sum()中没有参数，而是去遍历.前的数组中的内容。所以arguments在这里没用。因为arguments只能获得参数值，无法获得点前的数组。只有this，才能自动获得.前的对象。