1. 标签是不是最小的节点?

答: 一个<标签名>是一个标签，一个标签不会成为节点对象。只有由开始标签+结束标签联合表示的一个元素才是一个元素节点对象。

节点不分大小，只区分上下级。而元素节点不一定就是最底层的节点。还可能包含子元素或子文本。

但是，在元素树中，最底层节点一定都是元素节点。

2. <table class="data">

var table=document.getElementsByClassName("data");

table.getElementsByTagName("button") 报错！getElementsByTagName不是一个函数?

答: document.getElementById()只返回一个元素对象，所以可以直接使用这个元素对象，调用后续函数

但是其余三个getElementsByXxx()都返回的是集合。集合中装的才是找到元素对象。即使只找到一个元素，也是装在集合中返回的！如果要取出集合中唯一找到的这一个元素使用，应该:var 一个元素对象= getElementsByXxx() [0]

3. 为什么把<script>放在<head>中，或<body>开头就错了?

答: 网页是顺序执行。如果<script>放在网页内容的前边，则<script>中查找元素时，什么也找不到！

4. for(var btn of btns){  
 btn.onclick=function(){ btn.style.backgroundColor="red" }  
 }为什么无论点哪个按钮，总是最后一个按钮变红?

答: btn是循环变量，在循环过程中，绑定事件处理函数时，临时指一下当前元素。然后，btn会随着循环，向之后的元素移动。直到循环结束，btn只能指向最后一个按钮。

而事件处理函数虽然是在循环过程中绑定，但是并不是立刻执行！事件处理函数是在今后用户点击按钮时，才自动执行。而此时，循环早就结束了！btn只能是最后一个按钮。

所以在处理函数中不能用循环变量来当做当前元素使用的。应该用this->当前元素。因为时间发生时，浏览器自动执行事件处理函数的语句是: 当前元素.onclick(); this->当前元素。

5. for(var btn of btns){

btn.onclick=function(){

var btn=this; //可不可以去掉，因为外边已经有 变量btn了，为什么不能直接用？是否冲突？

btn.style.backgroundColor="red";

}

}

答: var btn=this;是必须的！因为外边的btn是全局的！极容易被污染！而之所以在函数内var btn，就是想在函数内定义一个专属的局部变量btn，不受外部btn的影响。名字相同，纯属巧合！

6. if(btn.innerHTML=="+"){//如果点的是+

n++;//就+1

}else if(n>1){//否则 如果点的是-，且内容>1

//才能-1

} else if中为什么不写btn.innerHTML=="-" ？

答: 一共只有两种按钮，+ 或 -。如果程序能执行到else if，说明前一个条件已经不满足了，就暗含着btn.innerHTML一定不是"+"，那就只能是"-"

7. var n=span.innerHTML，修改n，为什么页面上的span不变？

答: 按值传递。页面上span.innerHTML是一个原始类型的值——"1"，原始类型的值赋值给另一个变量时，是复制副本给对方。修改副本，不影响原值！所以，要想改变页面内容，必须再赋值回去！

8. 页面上span的内容显示没有"1"，只是一个数字1，为什么还要强制转换？

答: 凡是从页面上获得的，哪怕看着是数字的！其实一律都是字符串！因为页面上只能显示字符串！在算术计算前，都要强制转为数字，才能用！

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

btns[i].onclick=function(){

btns[i].backgroundColor=”red”;

}

}为什么也不对？

答：i是循环变量，i只有在遍历过程中，才临时指向某一个具体按钮在集合中的下标。而在遍历过程中，仅仅是对按钮的onclick属性赋值一个事件处理函数，并没有调用执行这个函数。因为事件处理函数，在任何情况下只有单击才执行。只要没单击，就不执行，也不会读取函数中的内容。函数中的i，只能是i，不会变成具体数组。

而当循环结束后，整段程序结束后，用户才单击按钮。单击按钮时，才开始读取处理函数的内容，而此时读到btns[i], 根据最基本的循环的原理，i已经变成刚好等于btns.length了。而btns中是没有btns.length这个下标的。Btns中的下标，最多才到btns.length-1。所以btns[i]返回undefined，所以undefined后边只要加.就会报错！

10. 事件处理函数为什么不能简化为箭头函数？

答：事件处理函数中的this，要求必须和外边的this不一样！所以，不能简化为箭头函数。如果非要简化箭头函数，也有一个比较别扭的方法:

btn.onclick=e=>{

var btn=e.target;

。。。 。。。

}

其中, e.target多数情况下等效于this。但是有不同的时候。所以，也只是临时的做法。

11. <span>1</span>这个1，到底是保存在哪里了？内存？网页上？

答：先写在程序中，<span>1</span>

再被浏览器解析到DOM树上，保存在对象属性中：span:{ nodeName: “SPAN”, innerHTML: “1” , parentNode: td, … }

最后，被浏览器画成界面，显示在网页中: 1

12. 为什么var n=parseInt(span.innerHTML), n++, 就能计算，而span++，就不行？

答: 因为span是一个元素对象，元素的内容是保存在对象的属性中的值。

所以，可以对元素对象中的某个属性+1，但是不能对元素对象整体+1。因为元素对象中包含很多属性值，到底要对那个属性+1呢？无法确定。

就好像，可以lilei.sage++，但是不能对Lilei++一样，没意义。你到底是想给lilei的身高+1，还是给lilei的年龄+1呀？

13. Uncaught DOMException: Failed to execute 'querySelectorAll' on 'Element': 'tbody td:last-chlid' is not a valid selector. ?

答: 'tbody td:last-chlid' 不是一个可用的选择器

凡是看到这个错，都是选择器写错了。

14. Uncaught DOMException: Failed to set the 'display' property on 'CSSStyleDeclaration': These styles are computed, and therefore the 'display' property is read-only.

答: 计算后的所有属性都是只读的。因为来源不确定！怕牵一发而动全身！所以，禁止修改！

总结: 今后，只要修改样式: 元素.style.css属性="值"

今后，只有获取样式时:

var style=getComputedStyle(元素);

style.css属性 ——只读

15. 有id的元素，不用查找，就可直接用id作为变量名，使用该元素，这样不是更简单吗？

答：浏览器确实可以将所有id属性值变为全局变量，并指向该元素对象：

<button id="btn1">

btn1.onclick=function(){ ... }

这里的btn1名，不用查找，也可以使用！而且，btn1指的就是<button>对象。这是浏览器自动转换的一个小福利。

但是，因为浏览器自动将id转为的是全局变量，非常不靠谱，极容易被篡改，所以，禁止直接使用id名来操作元素。所有元素使用之前，必须去DOM上根据特征或选择器找到，才更稳妥。

16. var style=getComputedStyle("d1")，程序出错了

报错! Uncaught TypeError: Failed to execute 'getComputedStyle' on 'Window': parameter 1 is not of type 'Element'

答: getComputedStyle()有规定，参数必须是一个实实在在的DOM元素对象！不能只是一个id名或class名

17. ，四个style不是说变量可以随意改名吗？为什么都改成style\_d1，却报错:

不能访问undefined的display属性?

答: 变量可以随意改名，所以前两个自己var出来的style可以改名： var style\_d1=getComputeStyle(d1)

if(style\_d1.display==="block")

但是，访问对象的属性，不能随便改名，因为属性时在对象中，提前定义好的！我们必须遵照执行

比如: 后两个d1.style，是访问DOM元素中专门保存内联样式的固定属性"style"。是DOM标准中规定的，必须叫这个名字。