**一、Js面向对象的几种方式**

1.对象的字面量 var obj = {}

2.创建实例对象 var obj = new Object();

3.构造函数模式 function fn(){} , new fn();

4.工厂模式：用一个函数，通过传递参数返回对象。function fn(params){var obj =new Object();obj.params = params; return obj;},fn(params);

5.原型模式：function clock(hour){} fn.prototype.hour = 0; new clock();

首先，每个函数都有一个prototype(原型)属性，这个指针指向的就是clock.prototype对象。而这个原型对象在默认的时候有一个属性constructor，指向clock，这个属性可读可写。而当我们在实例化一个对象的时候，实例newClock除了具有构造函数定义的属性和方法外（注意，只是构造函数中的）,还有一个指向构造函数的原型的指针，ECMAScript管他叫[[prototype]]，这样实例化对象的时候，原型对象的方法并没有在某个具体的实例中，因为原型没有被实例

**二、display：none和visibility：hidden的区别？**

Display：none 隐藏对应的元素，在文档布局中不再给它分配空间，它各边的元素会合拢，就当他从来不存在

Visibility：hidden隐藏对应的元素，但是在文档布局中仍保留原来的空间

**三、事件的三个处理过程？**

捕获：当我们在 DOM 树的某个节点发生了一些操作（例如单击、鼠标移动上去），就会有一个事件发射过去。这个事件从 Window 发出，不断经过下级节点直到触发的目标节点。在到达目标节点之前的过程，就是捕获阶段（Capture Phase）。（所有经过的节点，都会触发这个事件。捕获阶段的任务就是建立这个事件传递路线，以便后面冒泡阶段顺着这条路线返回 Window。）

目标阶段：当事件不断的传递直到目标节点的时候，最终在目标节点上触发这个事件，就是目标阶段。

冒泡阶段：事件冒泡即事件开始时，由最具体的元素接收（也就是事件发生所在的节点），然后逐级传播到较为不具体的节点（我们平时用的事件绑定就是利用的事件冒泡的原理）

阻止事件冒泡：

e. stopPropagation();//标准浏览器

event.canceBubble=true;//ie9之前

阻止默认事件：

为了不让a点击之后跳转，我们就要给他的点击事件进行阻止

return false

e.preventDefault();

**四、http响应码，200，300，301，302，401，403，404，500，502**

200 成功

300 多种选择

301 永久重定向 请求的网页永久移动到新的位置

302 临时重定向 服务器从不同位置嫌相应请求

401 未授权，要求身份验证

403 拒绝访问

404 未找到

500 服务器内部错误

502 错误网关 服务器作为网关或代理，从上游服务器收到无效响应。

**五、现在有一个字符串，里面由数字，英文句号和逗号组成，请编写一个函数，将里面所有出现的数字进行排序。**

var a = '1312213,64437.34,.435,346.,'

var b = a.split('')

var c = []

for(var i in b){

var z = /\w/;

if(z.test(b[i])){

c.push(b[i])

}

}

c.sort((num1, num2)=>num1 – num2) //返回值为负数num1在前

console.log(c)

**六、 写一个获取非行间样式的函数**

**currentStyle属性和getComputedStyle属性不能设置属性,只能获取**

currentStyle:该属性只兼容IE,不兼容火狐和谷歌

getComputedStyle:该属性是兼容火狐谷歌,不兼容IE

Function getStyle(obj, attr){

If(obj.currentStyle){

//IE兼容

return obj.currentStyle[attr];

}else{

//火狐谷歌兼容

return window.getComputedStyle(obj,null)[attr];

}

}