1、下面哪些是 NodeJS 官方模块

A.Querystring

B.Request

C.Async

D.Dns

2、常用的 git 操作有

A.Add

B.Push

C.Mkdir

D.Fetch

E.Mv

F.Merge

G.Tag

3、下面说法正确的有

A.P 元素不能包含 div

B.Li 元素的祖先元素可能是 li，但氟元素不可能是 li

C.Domtree 的根节点是 body 元素

D.Body 内的元素的 offsetparent 一定存在

4、在文件/home/somebody/workspace/somemodule.js 中第一行引用了一个模块：require('othermodule')，请问 required 的查找模块的顺序

A./home/somebody/workspace/mode modules/othermodule/index.js

B./home/somebody/workspace/mode_modules/othermodule.Js

C.CORE MODULES named othermodule

D./home/somebody/mode_modules/othermodule/index.js

5、请填充代码，使 mySort()能使传入的参数按照从小到大的顺序显示出来。

```
function mySort( )   {
        var tags = new Array();//使用数组作为参数存储容器
        请补充你的代码
        return tags;//返回已经排序的数组
}
var result = mySort(50,11,16,32,24,99,57,100);/传入参数个数不确定
console.info(result);//显示结果
```

6、 请写出个人 github 地址

7、请使用原生 js 实现一个 div 可拖拽，需要考虑浏览器兼容性。

8、如何判断浏览器是 IE 还是火狐，用 ajax 实现。要想通过 Ajax 来判断是 ie 浏览器还是 firefox 浏览器，就应该通过 XMLHttpRequest 对象。

热心网友答案详解（供参考），欢迎各路大神 PK 

1 解:在 stackoverflow 找到了一个比较合理的解释 .所以这一题我选了 A D。

2 解：对于这里的关键词“常用”，也是没有一个明显的界限的，你要是用的多，就叫常用。下面非别分析：

A： add：将当前工作目录中更改或者新增的文件加入到 Git 的索引中，加入到 Git 的索引中就表示记入了版本历史中，这也是提交之前所需要执行的一步。

B: push：将本地 commit 的代码更新到远程版本库中，例
如 “git push origin”就会将本地的代码更新到名为 orgin 的远程版本库中。

C：mkdir：应该不属于 git 常用操作的范围。

D：fetch：从服务器的仓库中下载代码。（与服务器交互，从服务器上下载最新代码）

E: mv： 重命名一个文件、目录或者链接。

F：merge：把服务器上下载下来的代码和本地代码合并。或者进行分支合并。

G：tag： 创建、列出、删除或者验证一个标签对象（使用 GPG 签名的）。

所以这一题应该是选：A B C E F G

3 解：A 肯定对，其它自己查。 

4 解：首先，nodejs 查找模块的方式与 Javascript 原型链或者作用域链的方式很相似。答案是：A B D C （很不确定）

5 解：这一题相对简单，是一道水题，直接上代码：

```
function mySort() {
        var tags = new Array();
        for(var i = 0;i < arguments.length;i++) {
              tags.push(arguments);
        }
       tags.sort(function(compare1,compare2) {
       return compare1 - compare2;
       });
       return tags;
}
var result = mySort(50,11,16,32,24,99,57,100);
console.info(result)
```

6 解：https://github.com/yuanzm7 解：如代码所示：

```
1 <html>
2 <head>
3     <title>test</title>
 4 </head>
5 <style type="text/css">
6 #drag1 {
7     width: 50px;
8     height: 50px;
9     background-color: #404040;
10     cursor: pointer;
11 }
12 </style>
13 <body>
14     <div id = "drag1"></div>
15 </body>
16 <script type="text/javascript">
17 window.onload = function() {
18     function Drag(obj) {
```

```
19        this.obj = obj;
20    }
21    Drag.prototype = {
22        constructor: Drag,
23        getInitPosition: function(e) {
24            e = e || window.event;
25            var eX,eY;
26            if(e.pageX || e.pageY){
27                eX = e.pageX;
28                eY = e.pageY;
29            }
30            eX = e.clientX;
31            eY = e.clientY;
32            var positionX = eX- this.obj.offsetLeft;
33            var positionY = eY - this.obj.offsetTop;
34            return {
35                x: positionX,
36                y: positionY
37            }
```

```
38        },
39        getmouseCoordinate:function(e) {
40            e = e || window.event;
41            if(e.pageX || e.pageY){
42                return {x:e.pageX, y:e.pageY};
43            }
44            return {
45                x:e.clientX + document.body.scrollLeft - document.
body.clientLeft,
46                y:e.clientY + document.body.scrollTop  - document
.body.clientTop
47             };
48        },
49        initDrag:function() {
50            var tempThis = this;
51            this.obj.onmousedown = function(e) {
52                var initP = tempThis.getInitPosition();
53                document.onmousemove = function(e) {
54                    var moveP = tempThis.getmouseCoordinate();
```

```
55                  tempThis.obj.style.marginTop = moveP.y - initP.
y + "px";
56                  tempThis.obj.style.marginLeft = moveP.x - initP.
x + "px";
57              }
58              document.onmouseup = function(){
59                  document.onmousemove = null;
60                  document.onmouseup = null;
61              }
62          }
63      }
64  }
65  var drag = document.getElementById("drag1");
66  var dragElement = new Drag(drag);
67  dragElement.initDrag();
68 }
69 </script>
70 </html>
```

提示：运行代码复制代码保存代码时，可以先修改部分代码再运行!

8 解：首先简单介绍一下这个对象：

(1)所有现代浏览器均支持 XMLHttpRequest 对象（IE5 和 IE6 使用 ActiveXObject）。

(2)所有现代浏览器（IE7+、Firefox、Chrome、Safari 以及 Opera）均内建 XMLHttpRequest 对象。

因此作者对于这题的理解是写一个基于 XMLHttpRequest 的 js 脚本，在 ie 或者火狐浏览器下面**判断到底处于哪一种浏览器环境,代码如下所示：**

```
var xmlhttp;
if (window.XMLHttpRequest) {
    // code for IE7+, Firefox, Chrome, Opera, Safari
   xmlhttp = new XMLHttpRequest();
   alert("your brower is not IE ");
} else {
   // code for IE6, IE5
  xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
  alert("your brower is IE ")
}
```