一、基础知识填空

JS中的数据类型包含哪些(要求详细)以及基本数据类型和引用数据类型的本质区别：

**基本数据类型**：number、string、boolean、null、undefined。

**引用数据类型**：数组[]、对象{}、正则/^\* /、function。

**区别**：基本数据类型是直接把值赋给变量、而引用数据类型是把对象的地址赋给变量。

JS中如何的检测数据类型(写出四种方法)：

typeof、instanceof、constructor、 Object.prototype.toString.call。

JS中判断某个值是否是一个有效数字的方法是：isNaN

想要接收函数执行的时候传递的参数，都可以如何的处理：*形参、arguments*

*将其他的数据类型转化为number类型的方式有那些：Number()，parseInt()，parseFloa*t()

二、写出DOM中常用的方法和属性

有一个div:<div class=”w”name=”h”id=”div1”></div>我想获取这一个div你有几种解决办法(不考虑兼容)：

document.getElementsByClassName(“w”)[0]、 document.getElementsByName(“h”)[0]、 doucunment.getElementById(“div1”)、 document.querySelector(“#div1”)、document.querySelector(“.w”)[0]、 document.querySelector(“div”)[0]

获取当前浏览器屏幕的宽度和高度(兼容所有的浏览器)： document.documentElement.clientWidth、 document.document.clientHeight

获取上一个哥哥元素节点（兼容所有的浏览器）:

function prev(curEle){

If(curEle.previousElementSibling){ return curEle.previousElementSibiling;

}

Var pre=curEle.previousSibiling;

While(pre&&pre.nodetype!==1){

Pre=pre.previousSibling;

}

Return pre;

}

动态创建一个div标签，并且添加到body的最后面位置：

var oDiv=doucument.createElement(“div”);

document.body.appendChild(“oDiv”);

继续上一题，把刚才创建的克隆一份一模一样的，添加到刚才创建的div前面：

Var oDiv2=oDiv.cloneNode(true);

document.body.insertBefore(oDiv2,oDiv)

三、数组的基础知识和应用

我想删除数组中的最后一项，至少写出两种办法：ary.pop()、ary.splice(ary.length-1,1)、ary.length--

我想在数组的末尾增加一个元素，至少写出两种办法：ary.push()、ary.splice(ary.length,0,x)、 ary[ary.length]=x

基于原来的数组复制一份一模一样新的数组，不考虑原有数组是否改变的情况，请写出至少三种方案：

Ary.slice()、ary.slice(0)、ary.concat()、ary.splice(0);

数组从小到大排序(可以处理10以上的数字)：ary.sort(function(a,b){return a-b;}) ;

实现找到第n项到第m项的内容，返回一个新的数组(原有数组不变)：slice(n-1,m);

四、写出下面表达式的运算结果

[]==false：true ![]==false：true []==![]：true Number(“13px”)==NaN：false

5+”3”-2：51

If(“3px”\*3){alert(“true”)}else{alert(“false”)}：”false”

If(“3px”+3){alert(“true”)}else{alert(“false”)}：”true”

If（document.body）{alert(“ok”);}else{alert(“no”)}：”ok”

alert([])：””

null==undefined：true

1. <input type="button" value="按钮1"/><input type="button" value="按钮2"/><input type="button" value="按钮3"/><input type="button" value="按钮4"/><input type="button" value="按钮5"/>

var ary=[“red”,”yellow”,”pink”,”green”,”blue”];

以上是页面中的5个按钮，和一个存储五个颜色值的数组，要求实现一个方法bindEvent，循环给5个按钮绑定点击事件，当点击的时候，整个body的背景颜色正好可以和数组中的指定项颜色对应(例如：点击第三个按钮，body的背景颜色”pink”)，请按照你的思路来实现?

var ary[**"red"**,**"yellow"**,**"blue"**,**"orange"**,**"purple"**]  
var oDiv=document.getElementById(**"div1"**).getElementsByTagName(**"input"**)  
for (var i=0;i<oDiv.length.i++){  
 oDiv[i].style.backgroundColor=ary[i];  
  
 oDiv[i].Q=i;  
 oDiv[i]onclick=function(){  
 document.body.style.backgroundColor=ary[this.Q]  
 }  
}

六、

var ary=[1,2,3,4];

function fn(ary){

ary[0]=0; -----> 把外边的那个传进来的ary的索引为0的那一项改成0；

ary=[0]; -----> 新建一个ary，里面只放了以个元素0；

ary[0]=100; ----->把新建的那个ary的索引为0的那一项改为100；

return ary; ----->返回上一步建立的ary；

}

var res=fn(ary); ---->定义一个变量res，并把fn执行后的返回值赋给res；

console.log(ary); ---->由于函数的执行，已经把全局作用域下的ary的内容改变了，第一项改成0；

故输出[0,2,3,4];

console.log(res); ---->返回值是函数里边新建的那个ary，故输出[100];

//此题偏难，认真思考(360面试题难度降低版本代码)

### 二、选择题：（要求写出分析步骤/画图）

**1、** console.log(a);

var a=12;

function fn(){

console.log(a);

var a=13;

}

fn();

console.log(a);

输出的三次分别是多少

1. JS代码运行之前，首先全局预解释：var a只声明，fn声明加定义；
2. 代码自上而下执行是。输出a 还未赋值 只是提前声明，所以 输出的是undefined，
3. 因为fn代码运行之前已经定义，所以直接跳过定义部分，当fn（）运行时形成一个私有作用域，

无形参所以不用赋值，预解释->var a提前声明 则为私有变量不受外界干扰；代码执行输出a时，此时私有变量a并未赋值，a只是声明。所以输出undefined；

1. 最后一次在全局输出a时，全局变量的a已赋值为12，所以输出12

选择B

A、undefined 12 13 B、undefined undefined 12

C、undefined undefined 13 D、有程序报错

1. console.log(a);

var a=12;

function fn(){

console.log(a);

a=13;

}

fn();

console.log(a);

1. JS代码运行之前，首先全局预解释：var a只声明，fn声明加定义；
2. 代码自上而下执行是。在全局输出的a 还未赋值 只是提前声明，所以 输出的是undefined，
3. 当fn（）执行时，形成的私有作用域时无形参，也无声明变量，代码执行 输出a时由于无私有变量则向上一级查找，在上一级window中var a =12，所以此时会输出12；
4. 代码继续执行时遇到a=13，由于也不是私有变量，依然像上一级查找变量 在window中找到var a=12；

便把全局变量的a的值修改为13；最后在全局输出的a便会为13；

所以选择A；

A、undefined 12 13 B、undefined undefined 12

C、undefined undefined 13 D、有程序报错

1. console.log(a);

a=12;

function fn(){

console.log(a);

a=13;

}

fn();

console.log(a); 输出的三次分别是多少

当代码在第一次执行输出a时；由于输出的a只是全局的一个属性名并不是变量，所以在全局不用参加预解释，也就不会声明，当执行时便会报错，a is not defined，报错后代码也不会往下执行了；

选择D.

A、undefined 12 13 B、undefined undefined 12

C、undefined undefined 13 D、有程序报错

1. var foo=1;   
    function bar(){

if(!foo){

var foo=10;

}  
 console.log(foo);

}  
 bar(); 输出的结果是多少

1. 首先全局进行预解释var foo声明，函数bar声明加定义；

代码自上而下执行时，给全局foo赋值为1，跳过bar已定义部分，然后bar（）执行时，形成一个私有作用域，由于不管if条件是否成立都要先进行预解释，在bar 执行时var foo 先进行预解释， foo为函数的私有变量，

1. 在if条件中，Foo转化为boolean类型并取反，由于判断是的foo预解释时只声明，并未定义，故foo为undefined，Undefined在boolean转化中为false，取反后为true，所以条件成立便给foo赋值为10，
2. 最后在bar私有作用域中输出的foo，为bar私有变量 便会输出10；

选择B

A、1 B、10 C、undefined D、报错

1. var n=0;

function a(){

var n=10;

function b(){

n++;

alert(n);

}

b();

return b;

}

var c=a();

c();

alert(n);

弹出三次的结果分别是什么？选择C

1. 全局预解释var n 和var c 只声明 ，函数a声明加定义，从上到下执行，全局n赋值为0；
2. 函数a定义部分跳过，a()执行返回结果变量c接收，函数a先执行形成一个私有作用域，无形参，

预解释 var n，函数b ，代码开始执行，私有变量n赋值为10，函数b定义部分跳过，b()执行，

b又形成一个私有作用域，无形参，无私有变量，执行到n++在上一级查找到变量n=10，n++=11；

再alert（n）->11.

1. 最后return b;把b的空间地址返回到外面并赋给全局变量c（var c=a（））；c()执行，也就是函数b在执行一次，又形成一个私有作用域，无形参，无私有变量 ，根据它在哪定义的找到上一级，此时的n在上次a（）执行完一次后，n的值为11；在次n++后也就是12，再次alert（n）->12；
2. 最后全局中alert（n）和函数里没有半毛钱关系，输出的已赋值完的全局变量n->0

选择C

A、1 1 1 B、11 11 0 C、11 12 0 D、11 12 12

1. var a=10,b=11,c=12;

function test(a){

a=1;var b=2;c=3;

}

test(10);

alert(a); alert(b); alert(c); 弹出的三个值分别是多少？

选择D;

1. 全局预解释var a为全局变量，函数test 声明加定义；b，c只是全局的属性名不预解释，
2. 代码自上而下执行，var a=10；设置个b，c两个属性名分别赋值b=10，c=12；
3. 跳过test定义部分，test()，test执行 形成一个私有作用域，有形参a并赋值为10，var b进行预解释-两个私有变量a，b。
4. 函数代码执行把形参a值修改为1；私有变量b赋值为2，c不是形参也不是私有变量，上一级查找也不是全局变量，最后找到全局属性c并把属性c的值修改为3；
5. Test （）中并没有返回值，所以alert(a)为全局变量输出->10; alert(b)全局属性->11; alert(c)全局属性值在函数中修改为3所以输出->3;

A、1 11 3 B、10 11 12 C、1 2 3 D、10 11 3

1. if(!("a" in window)){
2. var a=1;

}

alert(a);

选择B

分析：首先不管if条件是否成立都要先进行预解释，var a 提前声明，为全局变量 ，所有全局变量都属于window，

所以“a”in window 为true，！取反false，if条件不成立也就不给变量a赋值为1；所以alert a为默认值

undefined

所以选择B

弹出的a的值是？

A、1 B、undefined C、报错 D、以上答案都不对

8、var a=4;

function b(x,y,a) {

alert(a);

arguments[2]=10;

alert(a);

}

a=b(1,2,3); alert(a); 三次弹出的结果是

选择D

分析：1.首先全局预解释，var a ，函数b声明加定义；

2.代码执行 全局变量a赋值为4；跳过b定义部分，把函数b执行后的返回值赋给全局属性名a，

3．函数 b执行形成一个私有作用域。形参赋值x=1，y=2，a=3，xya为私有变量，无预解释，代码执行

alert（a）此时输出的->3, arguments[2]=10是把接收参数结合吧索引为2的值修改为10，私有变量a的值为10;再一次alert（a）->输出的是10；函数执行完 但并没有return返回值 ，test立即销毁；

1. a=b（1,2,3）指的是上面全局变量a把值修改为函数b的返回值，因为a既是全局变量也是全局属性名，
2. 但函数test并没有返回值所以alert（a）->undefined;

A、3 3 4 B、3 10 4 C、3 10 10 D、3 10 undefined

9、var foo='hello';

(function(foo){

console.log(foo);

var foo=foo||'world';

console.log(foo);

})(foo);

console.log(foo); 三次分别输出什么？

选择A，

1.全局预解释var foo；代码执行 全局变量foo赋值为字符串‘hello’，

2.自定义函数自执行；形成一个私有作用域，形参foo赋值为字符串‘hello’，预解释var foo；

代码执行第一次输出foo->给形参赋的-hello；给var foo这个私有变量赋值->var foo=foo||'world';

(如果形参foo的值不是undefined就把形参的值的给声明的私有变量foo，否则就把world的这个字符串赋给声明的私有变量foo)

3.第二次输出foo也就是刚赋值完的私有变量的值也就是->hello

4.第三次全局输出和闭包没有关系，输出的是全局变量foo的值->hello

三次输出的都是hello；选择A

A、hello hello hello B、undefined world hello C、hello world world D、以上答案都不正确

10、var a=9;

function fn(){

a=0; ->不是私有变量到上一级找到全局变量a，并把a的值改为0；

return function(b){ return b+a++; } ->a++ a和b先运算，运算完后a自身加1；

} ->return把这个小函数当做值返回给fn（）

var f=fn() ->把这个小函数空间地址赋给f

var m=f(5);alert(m); ->全局变量m接收小函数执行后返回值；f传进去一个5赋值给形参b，b+a++->

结果为5，并赋值给m这个变量。输出m->5

A0（fn（））栈内存被全局变量f引用所以不销毁，此时a自身加1->a=1，；

var n=fn()(5);alert(n); ->fn（）在执行一次 重新开辟一个A1栈内存把返回值赋给全局变量n；此时a找到全局变量a，并把a的值改为0；，然后小函数立即执行在里面也开辟了个空间，形参b赋值为5， b+a++;结果为5，a自身再次加1返回给n->输出n->5

var x=f(5);alert(x); ->小函数f再次执行传进去5，它的上一级依然是A0，此时A0的a的值为上一次执行变为1； b+a++结果为6，a自身再次加1，返回值赋给全局变量x执行结束；

alert （x）->6

alert(a); ->弹出的四次结果？ 由于小函数最后一次执行结束后a自身加1了两次，所以alert（a）输出 ->2

选择D

A、6 6 7 2 B、5 6 7 3 C、5 5 6 3 D、以上答案都不正确

三、思考题：

1. document.parentNode的值是什么，为什么

document.parentNode的值是null，因为document已经是最大的节点，它没有父亲节点

1. document.parentnode的值是什么，为什么

document.parentnode的值是 undefined，因为小写的parentnode不是一个属性，之前也没有它的定义，故就没有这个属性，所以它的值就是默认的undefined。

1. prototype是谁的属性，它起什么作用？\_\_proto\_\_是谁的属性，它有什么作用？prototype和\_\_proto\_\_有什么联系？

Prototype是函数（类）的属性，本身是对象数据类型的，因此执行的时候，浏览器给函数（类）开辟了一个堆内存，并且赋给它一个天生的属性constructor，属性值为当前函数（类）本身。对于类来说，prototype上的属性都是实例的公有属性。

\_\_proto\_\_ 是每一个对象数据类型天生自带的一个属性，属性值是当前实例所属类的原形（prototype），通过 \_\_proto\_\_ ，我们可以找到该实例所属类的prototype上。

通过 \_\_proto\_\_ 找到prototype，同时 \_\_proto\_\_ 也是prototype一个属性。

1. hasOwnProperty和in都是用来判断对象的属性的，这两个是怎么用的，有什么区别？

In是用来判断一个属性是不是属于这个一个对象，比如：“a”in obj；它不分辨私有属性和公有属性

hasOwnProper是用来判断一个属性是否是一个对象的私有属性，比如obj.hasOwnProperty(“a”);

1. 什么是闭包，它有什么作用？

闭包就是一个私有作用域，用来保护私有变量不受外界影响的一种机制。