**Web前端第二阶段JS基础考试试卷**

**总分100分 考试时间：120分钟 考试形式：闭卷**

**一、选择题（每题1分，共20分）**

1．分析下段代码输出结果是\_\_\_\_\_B\_\_\_\_\_。

var arr = [2,3,4,5,6];

 var sum =0;

 for(var i=1;i < arr.length;i++) {

      sum +=arr[i]

}

  console.log(sum);

A．20     B．18     C．14     D．12

2、以下关于 Function函数对象的说法不正确的是（D ）

A．函数可有可无返回值

B．调用函数时传递的参数是实参

C．函数调用时候，可以用一个变量来接收函数的返回值

D．函数必须要有形式参数

3、以下代码运行的结果是输出(B )

    var a = b = 10;

    (function(){

        var a = b = 20

    })();

    console.log(b);

A．10     B．20     C．报错     D．undefined

4、以下代码运行后的结果是输出(  A )

  var a=new Object();

a.name=”admin”

console.log(a.name);

A．admin B．a    C．“admin”     D．[a]

5、在 JS 中，’1555’ + 3 的运行结果是(  C )

A．1558     B．1552     C．15553     D．1553

6、以下代码运行后弹出的结果是(  B )

    var a = 888;

    ++a;

    alert(a++);

A．888     B．889     C．890     D．891

7、下列的哪一个表达式将返回值为假（ B   ）

A．!(3<=1)

B．(4>=4)&&(5<=2)

C．(“a”==“a”)&&(“c”!=“d”)

D．(2<3)||(3<2)

8、执行下面代码y 的值为（ D   ）

var x = 1;

function fn(n){

n = n+1

};

y = fn(x);

A．2     B．1     C．3     D．undefined

9、[1,2,3,4].join("0").split("") 的执行结果是（ C   ）

A．"1,2,3,4"

B．[1,2,3,4]

C．[“1”,“0”,“2”,“0”,“3”,“0”,“4”]

D．"1,0,2,0,3,0,4"

10、下面代码的运行结果是：第一次弹1，那第二次弹（ D   ）

    function fn1() {

        alert(1);

    }

    alert( fn1() );

A．1

B．alert(1);

C．function fn1() { alert(1); }

D．undefined

11，以下代码运行后，结果为（ B   ）

    fn1();

    var fn1 = function(a){ alert(a); }

A．1     B．程序报错     C．alert(1);     D．undefined

12，var n = "miao wei ke tang".indexOf("wei")；n的值为：（ B   ）

A．-1     B．5     C．程序报错     D．-10

13，下面的描述中不正确的是（ C   ）

A．arguments 中保存了实际传入函数内的所有参数。

B．return 只能在函数内部使用。

C．setInterval(fn1,1000) 只会调用一次 fn1。

D．Date 对象的 getMonth() 获取到的值比实际月份小1。

14，下面的描述中正确的是（C ）

A．"==" 在比较过程中，不但会比较两边的值，还会比较两边的数据类型。

B．typeof null 的输出的结果是 ’null‘。

C．typeof function() {} 的输出的结果是 ’function‘。

D．字符串的 length 可以获取，也可以设置。

15，以下代码中，会出现什么结果：第一次弹1，第二次弹1，第三次弹（B ）

    function fn1() {

        var a = 0;

        function fn2() {  ++a;  alert(a); }

        return fn2;

    }

    fn1()();

    var newFn = fn1();

    newFn();

    newFn();

A．1     B．2     C．0     D．3

16，以下代码中，哪个是不能正确获取到DOM元素的（B ）

A．getElementById()

B．getElementByClass()

C．getElementsByTagName()

D．querySelectorAll()

17，以下代码中哪个用法是错误的（B ）

A．divEl.style = “color: red”

B．divEl.class= “active”

C．divEl.classList.add(‘active’)

D．divEl.style.cssText= “color: red;font-size:20px”

18，以下代码中监听事件错误的写法是（A ）

1. divEl.click = function() {}
2. divEl.onclick = function() {}

C．divEl.addEventListener(‘click’, function() {})

D．<div onclick=”alert(‘1’)”>我是box</div>

1. 看下面的代码，执行后依次输出是( D )。

async function async1 () {

console.log('async1 start')

await async2();

console.log('async1 end')

}

async function async2 () {

console.log('async2')

}

console.log('script start')

setTimeout(function () {

console.log('setTimeout')

}, 0)

async1();

new Promise (function (resolve) {

console.log('promise1')

resolve();

}).then (function () {

console.log('promise2')

})

console.log('script end')

A.script start 、setTimeout、async1 start、async2、promise1、script end、async1 end、promise2

B.script start 、async1 start、promise1、script end、async2、async1 end、promise2、setTimeout

C.script start 、async1 start、async2、promise1、async1 end、promise2、script end、setTimeout

D.script start 、async1 start、async2、promise1、script end、async1 end、promise2、setTimeout

20. 在JavaScript中，下列关于window对象方法的说法错误的是( C )

A.window对象包括location对象、history对象和docuement对象

B.window.onload( )方法中的代码会在页面加载完成后执行

C.window.open( )方法用于在当前浏览器窗口加载指定的URL

D.window.close( )方法用于关闭浏览器窗口

**二、填空题（每空1分，共20分）**

1．JavaScript 中 8 种基本的数据类型（1）。

Number String Boolean Undefined Null Object BigInt Symbol

1. typeof 'coder' 输出的结果为（2）。

‘string’

1. var a = ‘6’ / ‘2’, a的值为（3）。

3

1. var b = ‘123’ + ‘4’, b的值为（4）。

‘1234’

1. JavaScript中常见的内置类有哪些（5）。

Number String Boolean Symbol BigInt Math Array Date

1. 元素的标准Attribute有哪些（6）。

id class style type value src ....

1. 请列出常见的事件，至少4个（7）。

click focus blur input keydown

1. this的绑定有几种方式（8）。

默认、隐式、显示、new

1. 常见的包管理器有哪些（9）。

Npm Cnpm yarn pnpm ….

1. 请写出一个匹配手机号码的正则（10）。

/^[1][3,4,5,6,7,8][0-9]{9}$/

1. JavaScript中常见的两种引用数据类型分别是（11）。

Array Object

1. JavaScript可以通过（12）函数延迟指定时间后去执行一次某些代码程序。

setTimeout()

1. JavaScript中String可以通过（13）方法获取某个子字符串第一次出现索引的位置。

indexOf()

1. 通过JavaScript代码使网页后退的代码是（14）。

history.back()

1. 用JavaScript的代码来获取当前的年月日（15）。

var date = new Date(); date.getFullYear(); date.getMonth() ; date.getDate()

1. 获取input输入框的聚焦和失去焦点的事件分别为（16）。

Onfocus onblur

1. 请写出将该数(5499.9674)保留2位小数的代码（17）。

5499.9674.toFixed(2)

1. 请列出数组常用的方法，至少4个（18）。

push pop sort join map filter findIndex reduce reverse

1. 看下面代码，执行后依次输出为（19）。

var name = 'window'

var person1 = {

name: 'person1',

foo1: function () {

console.log(this.name)

},

foo2: () => console.log(this.name),

foo3: function () {

return function () {

console.log(this.name)

}

},

foo4: function () {

return () => {

console.log(this.name)

}

}

}

var person2 = { name: 'person2' }

// 答题：

person1.foo1(); // 隐式绑定: person1

person1.foo2(); // 上层作用域: window

person1.foo4.call(person2)(); // person2

person1.foo4().call(person2); // person1

1. 请说出ES6新增了哪些新特性，至少写5个（20）。

let / const / 解构 /字符串模板/箭头函数/symbol/Set/Map

**三、简答题（60分）**

**1.说出 == 和 === 的区别。(5分)**

答:

* 普通相等: ==
  + 在比较类型不相同的情况下, 会将运算元先转成Number的值, 再进行比较(即会进行隐式转换 ) (3分)
* 严格不等: ===
  + 在进行比较时不会做任何的类型转换.例如:在类型不同的情况下, 直接返回false.(2分)

1. **写出动态创建<div class=”title”>我是box</div>，并插入到body中的代码(5分)**

答：

var divEl = document.createElement(‘div’)

divEl.className=”title”

divEl.innerText = “我是box”

document.body.append(divEl) (5分)

1. **什么是值传递，什么引用传递？(5分)**

答：

* 值类型(原始类型):
  + 在变量中保存的是值本身,占据的空间是在栈内存中分配的
* 引用类型(对象类型):
  + 在变量中保存的是对象的“引用”,占据的空间是在堆内存中分配的

1. 值传递: 是将值类型传递给函数参数,函数内部对参数的改变不会影响函数外部的变量， 例如:

function foo(a) {  
 a = 200  
}  
var num = 100  
foo(num)   
console.log(num) // 100

2)引用传递: 是将引用类型传递给函数参数,函数参数保存的是对象的"引用",在函数内部修改对象的属性会影响函数外部的变量，例如:

function foo(a) {  
 a.name = "HTML"  
}  
​  
var obj = {  
 name: "JavaScript"  
}  
foo(obj)  
console.log(obj.name) // HTML

1. **说出call、apply、bind三个函数的异同？(5分)**

答：

相同点：它们都可以显示的给函数帮定this的指向

用法如下：

* apply

第一个参数: 绑定this

第二个参数: 传入额外的实参, 以数组的形式

* call

第一个参数: 绑定this

参数列表: 后续的参数以多参数的形式传递, 会作为实参

* bind方法会创建一个新的绑定函数 ( 即不希望obj对象身上有该函数 )

   var bar = foo.bind(obj)  
•    bar() // this -> obj

区别：

* call、apply和bind都可以改变函数的this指向
* call、apply和bind第一个参数的是this要指向的对象
* call、apply和bind都可以后续为函数传参，apply是将参数并成一个数组，call和bind是将参数依次列出
* call、apply都是直接调用，bind生成的this指向改变函数需要手动调用。

1. **什么是闭包？闭包有哪些应用场景？(5分)**

答：**什么是闭包？**

一个普通的函数function，如果它可以访问外层作用域的自由变量，那么这个函数和周围环境就是一个闭包。

从狭义的角度来说：JavaScript中一个函数，如果访问了外层作用域的变量，那么它是一个闭包

答：**应用场景？**

防抖 、节流 、立即执行函数 、组合函数等等

1. **说出localStorage和sessionStorage的区别？(5分)**

答：

LocalStorag: 提供一种永久性存储的方法 在网页关闭打开时 依然保留

SessionStorage: 会话存储 再关闭该网页时 存储的内容被清除

它们的区别:

localStorage 永久性存储,

SessionStorage在关闭当前页面时存储的内容就会失效

localStorage除非手动删除否则一直存在

SessionStorage只能被同一个窗口的同源页面共享

1. **说出从输入URL到页面渲染完成的整个过程？(10分)**

答：

* URL 输入
  + 检查输入的内容是否是一个合法的 URL 链接
  + 判断输入的 URL 是否完整, 如果不完整，浏览器可能会对域进行猜测，补全前缀或者后缀
  + 使用用户设置的默认搜索引擎来进行搜索
* DNS 解析
  + 浏览器不能直接通过域名找到对应的服务器 IP 地址，所以需要进行 DNS 解析，查找到对应的 IP 地址进行访问。
* 建立 TCP 连接
* 发送 HTTP / HTTPS 请求（建立 TLS 连接）
  + 向服务器 发起 TCP 连接请求
  + 当这个请求到达服务端后，通过 TCP 三次握手，建立 TCP 的连接。
    - 1.客户端发送 SYN 包到服务器，并进入 SYN\_SEND 状态，等待服务器确认
    - 2.服务器收到 SYN 包，必须确认客户的 SYN，同时自己也发送一个 SYN 包，此时服务器进入 SYN\_RECV 状态。
    - 客户端收到服务器的 SYN包，向服务器发送确认包，此包发送完毕，客户端和服务器进入 ESTABLISHED 状态，完成三次握手。
* 服务器响应请求
  + 当浏览器到 web 服务器的连接建立后，浏览器会发送一个初始的 HTTP GET 请求，请求目标通常是一个 HTML 文件。服务器收到请求后，将发回一个 HTTP 响应报文，内容包括相关响应头和 HTML 正文。
* 浏览器解析渲染页面
  + 处理 HTML 标记并构建 DOM 树。
  + 处理 CSS 标记并构建 CSSOM 树。
  + 将 DOM 与 CSSOM 合并成一个渲染树
  + 根据渲染树来布局，以计算每个节点的几何信息。
  + 将各个节点绘制到屏幕上。
* HTTP 请求结束，断开 TCP 连接.

1. **用ES6实现Student继承Person，每个类至少有一个属性和方法？(10分)**

答：

class Person {

constructor(name, age) {

this.name = name

this.age = age

}

running() { console.log(‘runing’) }

eating() { console.log(‘eating) }

}

class Student extends Person {

constructor(name, age, sno, score) {

super(name, age)

this.sno = sno

this.score = score

}

studying() { console.log(‘studying’) }

}

var stu = new Student()

1. **写出Promise基本使用的代码，至少包含then、catch和finally方法？(10分)**

答：

var promise = new Promise( (resolve, reject)=>{

setTimeout(()=>{

resolve(‘coder’)

})

} )

promise.then((res)=>{

console.log(res)

})

.catch((err)=>{

console.log(err)

})

.finally((err)=>{

console.log(err)

})