上下文规则1

函数的上下文由调用函数的方式决定

- 函数的上下文（this关键字）由调用函数的方式决定，function是“运行时上下文”策略
- 函数如果不调用，则不能确定函数的上下文

规则1

- 规则1：对象打点调用它的方法函数，则函数的上下文是这个打点的对象

```
对象.方法()
```

规则1题目举例 - 第1小题

```
function fn() {
    console.log(this.a + this.b);
}

var obj = {
    a: 66,
    b: 33,
    fn: fn
};

obj.fn();
```

99

构成对象.方法()的形式，适用规则1

规则1题目举例 - 第2小题

```
var obj1 = {
    a: 1,
    b: 2,
    fn: function () {
        console.log(this.a + this.b);
    }
};

var obj2 = {
    a: 3,
    b: 4,
    fn: obj1.fn
};

obj2.fn();
```

7

构成对象.方法()的形式，适用规则1

规则1题目举例 - 第3小题

```
function outer() {
    var a = 11;
    var b = 22;
    return {
        a: 33,
        b: 44,
        fn: function () {
            console.log(this.a + this.b);
        }
    };
}

outer().fn();
```

77

构成对象.方法()的形式，适用规则1

规则1题目举例 - 第4小题

```
function fun() {
    console.log(this.a + this.b);
}
var obj = {
    a: 1,
    b: 2,
    c: [{
        a: 3,
        b: 4,
        c: fun
    }]
};
var a = 5;
obj.c[0].c();
```

7

构成对象.方法()的形式，适用规则1

上下文规则2

规则2

- 规则2：圆括号直接调用函数，则函数的上下文是window对象

函数()

规则2题目举例 - 第1小题

```
var obj1 = {
    a: 1,
    b: 2,
    fn: function () {
        console.log(this.a + this.b);
    }
};

var a = 3;
var b = 4;

var fn = obj1.fn;
fn();
```

7

构成函数()的形式，适用规则2

规则2题目举例 - 第2小题

```
function fun() {
    return this.a + this.b;
}
var a = 1;
var b = 2;
var obj = {
    a: 3,
    b: fun(),
    fun: fun
};
var result = obj.fun();
console.log(result);
```

适用规则2

适用规则1

6

上下文规则3

规则3

- 规则3：数组（类数组对象）枚举出函数进行调用，上下文是这个数组（类数组对象）

```
数组[下标]()
```

规则3题目举例 - 第1小题

```
var arr = ['A', 'B', 'C', function () {
    console.log(this[0]);
}];

arr[3]();
```

适用规则3

"A"

类数组对象

- 什么是类数组对象：所有键名为自然数序列（从0开始），且有length属性的对象

- arguments对象是最常见的类数组对象，它是函数的实参列表

规则3题目举例 - 第2小题

```
function fun() {
    arguments[3]();
}

fun('A', 'B', 'C', function () {
    console.log(this[1]);
});
```

适用规则3

"B"

上下文规则4

规则4

- 规则4：IIFE中的函数，上下文是window对象

```
(function() {

})();
```

规则4题目举例

```
var a = 1;
var obj = {
    a: 2,
    fun: (function () {
        var a = this.a;
        return function () {
            console.log(a + this.a);
        }
    })()
};
obj.fun();
```

适用规则4

适用规则1

3

上下文规则5

规则5

- 规则5：定时器、延时器调用函数，上下文是window对象

```
setInterval(函数, 时间);
setTimeout(函数, 时间);
```

规则5题目举例

```
var obj = {
    a: 1,
    b: 2,
    fun: function () {
        console.log(this.a + this.b);
    }
}
var a = 3;
var b = 4;

setTimeout(obj.fun, 2000);
```

→ 适用规则5

7

规则5题目举例

```
var obj = {
    a: 1,
    b: 2,
    fun: function () {
        console.log(this.a + this.b);
    }
}
var a = 3;
var b = 4;
setTimeout(function() {
    obj.fun();
}, 2000);
```

适用规则1

3

上下文规则6

规则6

- 规则6：事件处理函数的上下文是绑定事件的DOM元素

```
DOM元素.onclick = function () {

};
```

规则6 - 小案例1

- 请实现效果：点击哪个盒子，哪个盒子就变红，要求使用同一个事件处理函数实现

规则6 - 小案例2

- 请实现效果：点击哪个盒子，哪个盒子在2000毫秒后就变红，要求使用同一个事件处理函数实现

谢谢！