1. 在比较判等的时候，应该尽量使用===运算符，双等于号会自动转换数据类型，有时会得到很诡异的结果。
2. Null表示一个空的值，他和0以及空字符串‘’不同，0是一个数值，‘’表示长度为零的字符串，而null表示“空”。
3. 与之相似的是undefined，它表示未定义，大多数情况下我们都应该使用null
4. 新的字符串拼接方式，var name=’小明’；

Var age=20;

Var message=’你好，${name},你今年${age}岁’

1. javaScript为字符串提供了一些常用方法，注意，调用这些方法本身不会改变原有的字符串内容，而是返回一个新的字符串。
2. **字符串操作**
3. toUpperCase();把一个字符串全部变成大写。
4. toLowerCase();把一个字符串全部变成小写。
5. indexOf(); 会搜索指定字符串出现的位置。
6. substring(0,5); 会从索引0开始到索引5结束，不包括索引5
7. **数组操作**
8. indexOf(); 搜索一个指定元素的位置
9. slice(); 对应sting的substring();，截取对应索引的元素
10. push(); 向数组的末尾添加若干元素。
11. pop(); 把数组的最后一个元素删除掉
12. unshift(); 向数组的开头添加元素，
13. shift(); 把数组的第一个元素删除
14. sort(); 对数组元素进行排序
15. reverse(); 反转数组
16. splice(); 从指定索引删除若干元素，然后添加
17. 例如arr.splice(2,3,’A’,’B’,’C’); 指从索引2删除3个元素，然后在再添加3个元素
18. concat(); 把当前的数组和另一个数组连接起来，并返回一个新的**数组**
19. join(); 把数组的每个元素都用指定的字符串连接起来，然后返回连接后的字符串

**对象操作**

1. js的对象是一种无序的集合数据类型，他由若干键值对组成。
2. javascript的对象，他的属性名必须是一个有效的变量名，如果属性名包含特殊字符，必须用’’括起来。例如，’middle-school’
3. 而且由于这个属性名，middle-school不是一个有效的变量，访问这个睡醒也无法使用.操作符，必须使用[‘xxx’]来访问。
4. 例如小明[‘moddle-school’];
5. 事实上，javascript对象的所有属性都是字符串，不过属性对应的值可以是任意的数据类型。
6. Javascript的对象是动态类型，你可以自由的对一个对象添加或删除属性。
7. 新增 xiaoming.age=18; 删除：delete xiaoming.school;
8. 可以使用hasOwnProperty(‘name’);判断属性是否是对象自身拥有的。

**条件判断**

1.if else if else

2.JavaScript 将null,undefined,0,NaN,和空字符串’’视为false,其他值一概视为ture。

**循环**

1. for循环
2. for…in 循环

例如 var o = {

name:’Jack’,

age:20,

city:’Beijing’

};

for (var key in o){

if(o.hasOwnProperty(key)){

console.log(key); ‘name’,’age’,’city’

}

}

1. 同样for …in循环可以直接循环出Array的索引
2. var a=[‘A’,’B’,’C’];

for (var i in a){

console.log(i); //’0’, ‘1’, ‘2’

console.log(a[i]); // ’A’,’B’,’C’

}

1. while循环只有一个判断条件，条件满足，就不断循环，条件不满足时退出循环，例如计算100以内所有奇数之和

var x=0;

var n=99;

while(n>0){

x=x+n;

n=n-2;

}

**map和set**

1. Map是一组键值对的结构，具有极快的查找速度。
2. 实例Map var m = new Map([[‘A’,95],[‘B’,75],[‘C’,85]]);

m.get(‘A’); //95

3.Map的操作方法 var m = new Map();

m.set(‘A’,67);

m.has(‘A’); //true

m.get(‘A’); //67

m.delete(‘A’);

4.由于key只能对应一个value，所以后面的值会覆盖前面的值。

5.Set和Map类似，也是一组key的集合，但是他不存储value，由于key不能重复，所以在Set中，没有重复的key

var s1=new Set();

var s2=new Set([1,2,3,4,5]);

重复的key会被过滤掉，可以使用add(key);添加元素到Set中，可以重复添加，但不会有效果。Delete(key); 删除元素。

**Iterable**

由于数组可以使用下标循环，遍历Map和Set就无法使用下标。为了统一集合类型，引入iterable类型。Array、Map、Set都属于iterable类型。可以使用for…of循环来遍历。

for …in 由于历史遗留问题，他遍历的其实是对象的属性名称，一个array数组实际上也是一个对象，他的每个元素的索引被视为一个属性。

当我们手动给Array对象添加额外的属性后，foo… in 循环将会带来意想不到的结果。

var a=[‘A’,’B’,’C’];

a.name=’Hello’;

for(var x in a){

console.log(x); //’1’,’2’,’3’,’name’

}

for…in循环将把name包括在内，但Array的length属性却不包括在内。

for…of循环则完全修复了这些问题，他只循环集合本身的元素。

var a=[‘A’,’B’,’C’];

a.name=’Hello’;

for(var x of a){

console.log(x); //’A’,’B’,’C’

}

最好使用iterable内置的forEach方法，他接受一个函数，每次迭代都自动回调该函数。

例如： var a=[‘A’,’B’,’C’];

a.forEach(function(element,index,array){

//element:指向当前元素的值

//index：指向当前索引

//array:指向array对象本身

Console.log(element+’,index=’+index);

});

结果： A，index=0

B，index=1

C，index=2

Set没有索引，因此回调函数的前两个参数都是元素本身