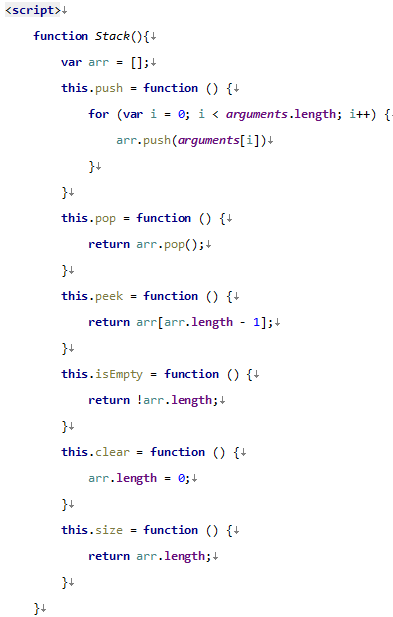
12.14-js09

*一、栈结构方法的封装*

1、封装

2、进制转换

}

*二、队列*

1、\* 队列：只允许在一端进行插入操作，在另一端进行删除操作。先进先出，后进后出。 \* 使用pop（）和unshift（）或者 使用 shift（）和push（）方法 都可以模拟队列

2、模拟队列题目：

2.1 把数组元素所有的值都放大10倍

2.2 踢猴子

*三、删除元素*

1、pop方法可以删除尾部元素，拎出末尾并返回。

2、shift可以删除头部元素。返回删除的元素，后边所有的元素全部向前移动1位，如果shift删除空数组，则不进行任何操作，返回undefined。

3、delete删除指定下标的元素，数组变成了空位数组 长度不变。

4、使用length属性也可以删除数组。empty \* n。

5、splice方法删除数组；返回被删除元素所组成的数组。

001、传入1个参数，代表从下标x始删（不包括x）到末尾。

002、传入2个参数，代表从下标x开始删（包括x） 总共删几个。可以不执行删除操作，即第2个参数是0，但是不能省略。

003、如果指定三个 或 多个参数，则第三个及后边所有的参数都是插入的元素，在第一个参数所指定的下标处开始插入。

004、当第一个参数大于等于数组长度的时候，删除无效，但是能在数组末尾添加新元素。

005、参数为负时，如果第一个参数为负数，则从右向左定位（从1开始数而不是0）,但是仍然是从左向右删除。第二个参数如果是负数，则被视为0。

 *四、添加元素*

1、push可以在数组末尾添加元素。返回新数组的长度。

2、unshift可以把1个或多个参数插入到数组的头部。多个参数时，第一个参数插在下标为0的位置，后边依次排列。 返回数组的新长度。注意：使用unshift依次插入和一次性插入是不一样的。

3、可以使用中括号和下标值 来添加元素，或者是修改元素。

4、concat方法能够把传递的参数按顺序依次插入到数组的末尾。可以拆散一维数组进行插入。不会修改原数组，而是返回一个拼接值以后的新数组。

5、splice方法可以在指定位置的下标添加元素。无论是否删除，都会把第三个参数及以后的所有参数按顺序放在第一 个参数指向的下标所在的位置及该位置之后的位置。返回被删除的元素所组成的新数组。

* 五、截取数组*

 1、splice方法

 2、slice(a,b)方法：该方法只有两个参数，截取并返回指定的区段。

001、参数a：代表起始下标（包含a）。为负则使用倒数定位法（参见splice方法）。

002、参数b：代表结束下标（不包含b）。为负则使用倒数定位法（参见splice方法）。

003、不传参则不做任何操作。

004、仅传一个参数则直接从该参数所指向的下标一直截取至数组的末尾。并返回。

005、若起始下标a >= 结束下标b，则无效。(即a不能位于b之后)

|  |  |  |
| --- | --- | --- |
|  | splice | slice |
| 返回 | 删除的数组 | 截取的数组 |
| 一个参数 | 下标x始删（不包括x）到末尾 | 从该参数所指向的下标一直截取至数组的末尾 |
| 传入2个参数（a,b） | 从下标a开始删（包括x） 总共删b个 | 代表起始下标（包含a），代表结束下标（不包含b）。为负则采用倒数定位法。 |
| 传入3个及以上参数（a,b，……） |  | ------ |
| 不传参 | 不做任何操作 | 不做任何操作 |
| 注 |  |  |

*六、数组倒序*

1、reverse方法，该方法没有参数，没有返回值，会改变原数组（像给一个变量重新赋值那样）。

2、手写reverse：

<script>

function reverse(arr){

var newArr = [];

for(var i in arr){

newArr.unshift(arr[i]); //循环将arr中的元素取出来放入newArr中下标为0的位置。unshift返回的是新数组的长度。

}

return newArr；

}

***console***.log(*reverse*([1, 4, 3, 2, 5, 1]));

</script>

 *七、数组排序sort方法*

1、无参，则先把每个元素转为string然后按照ASCII码顺序排序。

2、参数为一个函数：sort(function(a,b)，函数中传入两个值a（代表后面的值）、b（代表前面的值），意思是，拿a与b比较即为拿后面一个与前面一个比较。比较两个值的大小并返回一个说明大小关系的数字（负数：a与b交换位置；正数：位置不变；0）。

3、升序排列：

***arr***.sort(**function** (a,b) {  
 **return** a - b;*//升序排列*})

4、对对象进行排序：数组里存入的元素是对象，对这些对象的某个属性进行排序。如按手机价格降序。



5、价格降序优化一点点。



6、es6新增的字符拼接：${X}用来包含变量X，所在的字符串一定要用反引号，单引号无效。 反引号就是1左边的键。

var name=‘Jhon’;

console.log(My name is ${name}.);

输出结果：My name is Jhon.

* 八、数组转换*

1、toString()方法： \* 当数组调用toString方法的时候，把数组中的每一个元素都转成字符串，然后用逗号连接显示 \* 当数组被放置在字符串的环境中时，JS会自动调用toString方法把数组转成字符串。注意null和undefined没有toString()方法。 \* 如： var arr = [1,2,3]; console.log(arr + "1"); //1,2,31

2、toLocaleString方法：会为数组添加千位分割符，可转换日期等。

3、join()方法：在调用join方法的时候，可以传入一个参数，代表分割符的样式；如果省略参数，则默认分隔符是逗号。 \* join返回字符串，不改变原数组。 \* 如果不想要间隔，可以使用空字符串 \* 如果碰到null或undefined则转成空字符串，其他值按照toString的方法转。

*九、数组元素定位*

1、indexOf(要检索的元素，从哪个下标开始找)。返回某个元素在数组中第一次匹配到的位置所在的下标；如果没有找到指定的值，则返回-1。

2、lastIndexOf(要检索的元素，往后检索的范围)：返回指定的值在数组中最后一个匹配的项，用法和indexOf相同。

*十、晨测*

1、说出sort排序原理 及如何降序或升序

2、splice方法的作用

3、slice方法的使用

4、添加元素

5、删除元素

6、两种删除元素的区别

 7、.和[]的区别

|  |  |
| --- | --- |
| .点 | []中括号 |
| 1、 可以理解为是一个静态访问对象的属性（或者对象的方法） | 1、可以理解为动态的访问属性（或者方法） |
| 2、按变量名访问，比较单一，不可以修改 | 2、可以用赋予字符串的方式访问，灵活性更大，常见的是可以遍历对象的属性 |
| 3、对象的key 不是数字才可以使用点语法 | 3、对象的 key是数字时， obj[-1]; |

8、toString和join的区别