数组与字符串的常见操作:

1. `slice()`, 用于返回新数组或字符串

```
arr.slice(start,end)
```

start为切割的初始位置(必需), end为结束位置(不包含第end个)(可选, 若end未赋值,则从start到数组结束),示例:slice(0,2),从第0个开始, 第二个之前; 若start为负数, 则从数组末尾从后往前数,end同样为结束的位置.详情示例见下图.

```
var arr = ["0", "1", "2", "3", "4", "5"]
undefined
console.log(arr.slice(1,3))
▶ (2) ["1", "2"]
undefined
console.log(arr.slice(-3,4))
▶ ["3"]
```

注:只是从大数组中返回一个子数组, 并**不会改变原有数组/字符串**. 注意与`splice`的区别.

2. `splice()`, 用于切割数组

```
arr.splice(start, length, items)
```

start为开始的位置(必需), length是截取的位数(必需), items是新增的(可选),start若为负数则从末尾从后往前进行, *length>=0*.

```
var arr = ["0", "1", "2", "3", "4", "5"]
undefined
console.log(arr.splice(0,2))
▶ (2) ["0", "1"]
undefined
console.log(arr.splice(-1,1))
▶ ["5"]
```

注:返回的是**被切割下来**的部分, 会**改变原数组**

3. `substr()`, 用于从字符串提取字符

```
str.substr(start,length)
```

start为开始的位置(必需), length为提取的位数(可选), start若为负数, 则从末尾开始, 若length大于实际start开始到末尾的位数, 则与substr(start)作用相同.*length>0*. length若未赋值, 则提取内容为从start到最后.

```
var str = "hello world"
undefined
console.log(str.substr(3,4))
lo w
undefined
console.log(str.substr(-3,4))
rld
```

注:不会修改原字符串

4. `substring()`, 用于提取字符串中介于两个指定下标之间的字符.

```
str.substring(start, end)
```

start为起始位置, end为结束位置(可选), 若未赋值, 则从start的位置到字符串结束.

```
var str = "helloWorld"
undefined
console.log(str.substring(2,5))
llo
undefined
console.log(str.substring(4))
oWorld
```

注:不会改原字符串, **start不可为负数!**

5. `split()`, 用于把字符串分割成数组

```
str.split(separator,length)
```

separator为切割的依据, length为返回数组的最大长度.

```
var str = "hello world"
undefined
console.log(str.split("", 5))
▶ (5) ["h", "e", "l", "l", "o"]
undefined
console.log(str.split(" "))
▶ (2) ["hello", "world"]
```

```
var str = "总经理前置_决策类_20180220"
undefined
console.log(str.split("_"))
▶ (3) ["总经理前置", "决策类", "20180220"]
```

6. `join()`, 用于把数组中元素拼接为字符串

```
arr.join(separator)
```

separator指定的分隔符(可选).如为空, 则以逗号为分隔符.

```
var arr =  ["总经理前置", "决策类", "20180220"]
undefined
console.log(arr.join("o_o"))
总经理前置o_o决策类o_o20180220
```

7. `concat()`, 用于拼接字符串/数组

```
arr.concat() / str.concat()
```

```
var arr =  ["总经理前置", "决策类", "20180220"]
undefined
var ARR = ["第二个数组", "里的东西"]
undefined
console.log(arr.concat(ARR))
▶ (5) ["总经理前置", "决策类", "20180220", "第二个数组", "里的东西"]
undefined
var str = "结束了附件的路上解放了"
undefined
console.log(str.concat("啦啦啦啦啦"))
结束了附件的路上解放了啦啦啦啦啦
```

8. `toLowerCase()` / `toUpperCase()`, 用于转化字符串大小写

```
str.toLowerCase() / str.toUpperCase()
```

9. `charAt()`, 用于返回字符串对应位置字符

```
str.charAt()
```

```
var str = "我爱你"
undefined
console.log(str.charAt(1))
爱
```

10. `indexOf()`, 用于返回某个特定字符串在字符串中首次出现的位置

```
str.indexOf(searchvalue,fromindex)
```

searchvalue为索引的内容(必须), fromindex为开始索引的位置(可选),为空则从首字符开始索引. *如果要检索的字符串值没有出现，则该方法返回 -1。*

11. `lastIndexOf()`, 用于返回一个字符串最后出现的位置, 方法同上.

12. `test()`, 用于返回字符串是否匹配特定的模式.

```
str.test()
```

```
var str = "hello world"
undefined
console.log(/hello/.test(str))
```