

又又 拍 针

i, j 都从前向后扫

给定一个数组 `nums`，编写一个函数将所有 `0` 移动到数组的末尾，同时保持非零元素的相对顺序。

示例：

输入：[0,1,0,3,12]

输出：[1,3,12,0,0]

说明：

1. 必须在原数组上操作，不能拷贝额外的数组。
2. 尽量减少操作次数。

思路1: 能大概想到用指针，遇0移动？

eg1: [0, 1, 0, 3, 12]

① $\uparrow \uparrow$
 i j j

② j 遇到非0的，停； $a[i] = a[j]$ ； $i++$

③ j 遇到非0的，停； $a[i] = a[j]$ ； $i++$

④ j 遇到非0的，停； $a[i] = a[j]$ ； $i++$
 j 完结。

最后需要将 i 之后的数 = 0.

eg2: [2, 0, 3, 4, 0, 12]

① $\uparrow \uparrow$
 i j
第1个数不是0: $j++$

② j 遇到非0的2，停； $a[i] = a[j]$
(自操作)

while ($j < \text{num.length}$)

{ if ($\text{nums}[j] \neq 0$) { $\text{nums}[i++] = \text{nums}[j]$; }

$j++$;

}

while ($i < \text{num.length}$)

$\text{nums}[i++] = 0$;

思路: (优化的思路, j 结束之后, 加个 if , 有一个循环内解决的问题).

```
for (j=0; j<n; j++) {
```

```
    if (n[j] != 0) {
```

```
        n[i] = n[j];
```

//同上

```
        if (i != j) {
```

```
            n[j] = 0;
```

```
        }
```

```
        j++;
```

```
    }
```

```
}
```

} 等于在将前局的数搞完后,
直接将该局的数置为 0.

思路3 优化方法2: 交换 而不是 $a[i] = a[j]$

$i = 0$

for ($j = 0; j < \text{len}; j++$)

if ($a[j] != 0$) {

int temp = a[i]; // 暂存前面的0

$a[i] = a[j];$ // 前面的0直接赋值给后面的数

$a[j] = \text{temp};$ // 后面的数赋值给0

$i++;$

}

}