#LeetCode75

https://leetcode.com/problems/sort-colors/description/

Yifeng Zeng

#题目描述

Sort Colors

#思路报告

拿到这个题最primitive的想法肯定是自己写一个quick sort/merge sort甚至直接调用Arrays.sort()。但这并没有利用到只有0，1，2三种color的特质，所以quick sort/merge sort明显就不再是考点了。既然我们已经知道了有固定3种颜色，那么我们可以用三个变量直接存一下每个颜色出现了多少次然后把每个颜色出现的次数对应回原array就可以了，这样的话可以做到时间O(n)，空间O(3)的情况，但是说白了这个O(3)就是O(k),k是color的个数，而当k等于n时这是不太好的，那么我们可不可以省略这个O(k)的空间而保持O(n)的时间呢？在线性array上能做的不要extra space的inplace操作其实就只有swap，那么我们就考虑swap。而要O(n)的时间，肯定是多个（k个）指针，一个指针指向array里0在的那一整块，一个指针指向1的那一整块，一个指针指向2的那一整块。对于一个整块里面不属于它的element进行swap操作。

代码如下

```
    public void sortColors(int[] nums) {
        if (nums == null || nums.length <= 1) {
            return;
        }
        int i = 0;
        int l = 0;
        int r = nums.length - 1;
        while (l <= r) {
            if (nums[l] == 0) {
                nums[l] = nums[i];
                nums[i] = 0;
                i++;
                l++;
            } else if (nums[l] == 1) {
                l++;
            } else if (nums[l] == 2) {
```

```
                nums[l] = nums[r];
                nums[r] = 2;
                r--;
            }
        }
    }
```

#套路总结

- array的inplace操作只能做swap，这个挺重要的，要有这个敏感度。