

4月24日报

本日学习内容

1. 学习时空复杂度的分析，总结博客
2. 学习git 中checkout, branch, merge 等操作，在模拟网站练习


今日算法题

题目1: [2089. 找出数组排序后的目标下标](#)

2089. 找出数组排序后的目标下标

已解答 

简单

 相关标签

 相关企业

 提示

Aa

给你一个下标从 0 开始的整数数组 `nums` 以及一个目标元素 `target` 。

目标下标 是一个满足 `nums[i] == target` 的下标 `i` 。

将 `nums` 按 非递减 顺序排序后，返回由 `nums` 中目标下标组成的列表。如果不存在目标下标，返回一个 空 列表。返回的列表必须按 递增 顺序排列。

示例 1:

输入: `nums = [1,2,5,2,3]`, `target = 2`

输出: `[1,2]`

解释: 排序后, `nums` 变为 `[1,2,2,3,5]` 。

满足 `nums[i] == 2` 的下标是 1 和 2 。

示例 2:

输入: `nums = [1,2,5,2,3]`, `target = 3`

输出: `[3]`

解释: 排序后, `nums` 变为 `[1,2,2,3,5]` 。

满足 `nums[i] == 3` 的下标是 3 。

示例 3:

输入: `nums = [1,2,5,2,3]`, `target = 5`

输出: `[4]`

解释: 排序后, `nums` 变为 `[1,2,2,3,5]` 。

满足 `nums[i] == 5` 的下标是 4 。

```
class Solution {
public:
    vector<int> targetIndices(vector<int>& nums, int target) {
        int n = nums.size();
        sort(nums.begin(), nums.end());
        vector<int> res;
        for (int i = 0; i < n; ++i){
            if (nums[i] == target){
                res.push_back(i);
            }
        }
        return res;
    }
};
```

```
}  
};
```

题目2: [2606. 找到最大开销的子字符串](#)

已解答 

2606. 找到最大开销的子字符串

中等

 相关标签

 相关企业

 提示

Ax

给你一个字符串 `s`，一个字符 **互不相同** 的字符串 `chars` 和一个长度与 `chars` 相同的整数数组 `vals`。

子字符串的开销 是一个子字符串中所有字符对应价值之和。空字符串的开销是 `0`。

字符的价值 定义如下：

- 如果字符不在字符串 `chars` 中，那么它的价值是它在字母表中的位置（下标从 `1` 开始）。
 - 比方说，`'a'` 的价值为 `1`，`'b'` 的价值为 `2`，以此类推，`'z'` 的价值为 `26`。
- 否则，如果这个字符在 `chars` 中的位置为 `i`，那么它的价值就是 `vals[i]`。

请你返回字符串 `s` 的所有子字符串中的最大开销。

示例 1:

输入: `s = "adaa", chars = "d", vals = [-1000]`

输出: `2`

解释: 字符 `"a"` 和 `"d"` 的价值分别为 `1` 和 `-1000`。

最大开销子字符串是 `"aa"`，它的开销为 `1 + 1 = 2`。

`2` 是最大开销。

示例 2:

输入: `s = "abc", chars = "abc", vals = [-1,-1,-1]`

输出: `0`

解释: 字符 `"a"`，`"b"` 和 `"c"` 的价值分别为 `-1`，`-1` 和 `-1`。

最大开销子字符串是 `""`，它的开销为 `0`。

`0` 是最大开销。

```
class Solution {  
public:  
    int maximumCostSubstring(string s, string chars, vector<int>& vals) {
```

```
int mapping[26]{};
iota(mapping, mapping + 26, 1);
for (int i = 0; i < chars.length(); i++) {
    mapping[chars[i] - 'a'] = vals[i];
}

int ans = 0, f = 0;
for (char c : s) {
    f = max(f, 0) + mapping[c - 'a'];
    ans = max(ans, f);
}
return ans;
}
};
```

明日学习计划:

1. 开始学习oc面向对象部分
2. 每天坚持写hot100