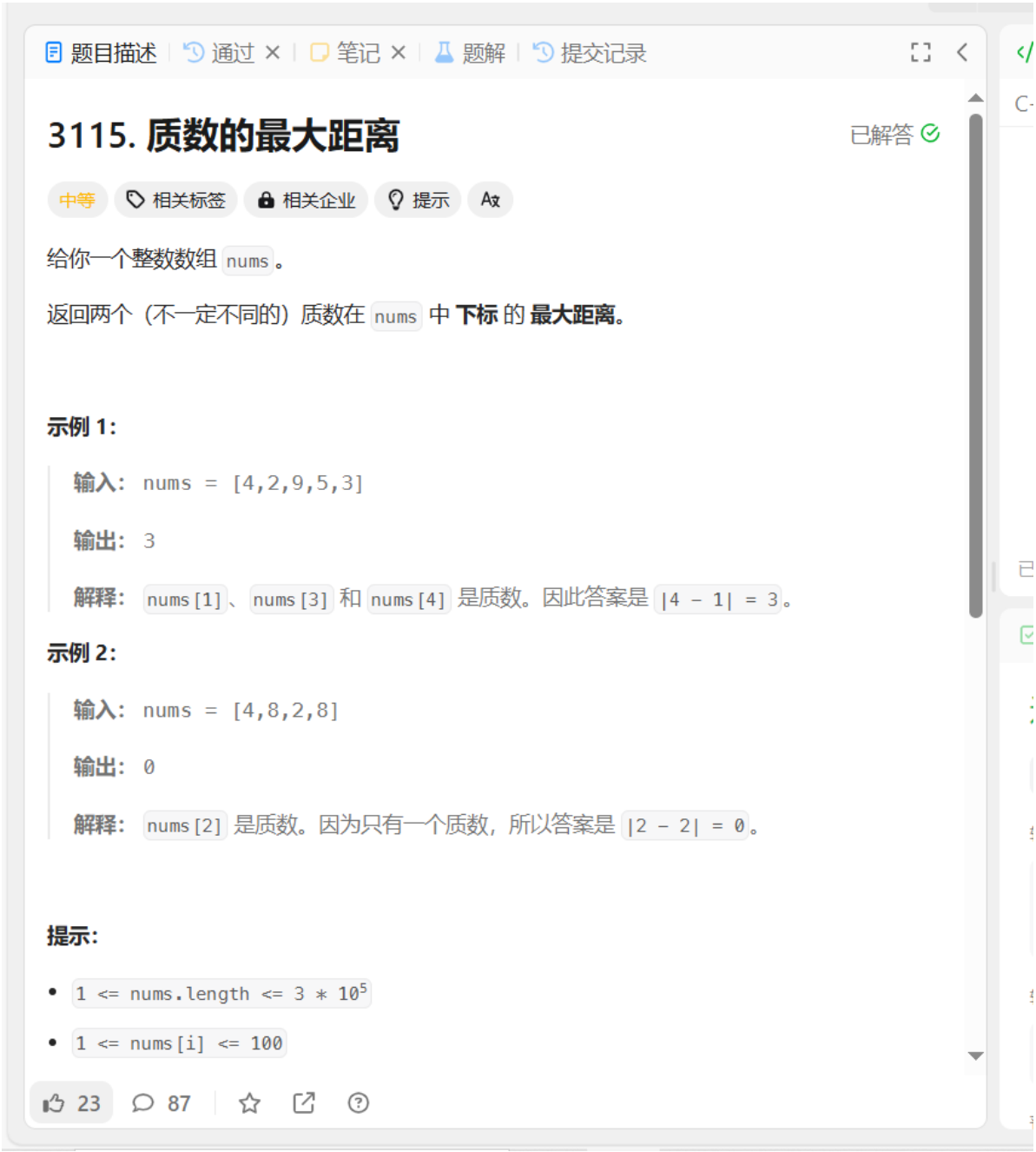
# 题目：质数的最大距离



题目描述 | 通过 × | 笔记 × | 题解 | 提交记录

## 3115. 质数的最大距离

已解答

中等 相关标签 相关企业 提示

给你一个整数数组 `nums`。

返回两个（不一定不同的）质数在 `nums` 中 **下标** 的 **最大距离**。

**示例 1：**

> **输入：** nums = [4,2,9,5,3]
>
> **输出：** 3
>
> **解释：** `nums[1]`、`nums[3]` 和 `nums[4]` 是质数。因此答案是 `|4 - 1| = 3`。

**示例 2：**

> **输入：** nums = [4,8,2,8]
>
> **输出：** 0
>
> **解释：** `nums[2]` 是质数。因为只有一个质数，所以答案是 `|2 - 2| = 0`。

**提示：**

- $1 <= nums.length <= 3 * 10^5$
- $1 <= nums[i] <= 100$

23 87

# 解题思路一：

保留每个质数的下标，最后把最后一个质数与第一个质数相减，得到最长距离

```
class Solution {
public:
    int maximumPrimeDifference(vector<int>& nums) {
        int lenth = nums.size();
        //初始化质数数组
        vector<int> result(lenth, -1);
        int last = 0;
        for (int i=0;i<lenth;i++)
        {
            //如果是质数
```

```
            if (isrightVa(nums[i]))
            {
                result[last] = i;
                last++;
            }
        }

        if (--last == 0)
            return 0;
        else
            return result[last] - result[0];
    }
private:
    bool isrightVa(int n)
    {
        if (n == 1) return false;
        if (n > 1) {
            for (int i = 2; i <= sqrt(n); i++)
            {
                if (n % i == 0)
                {
                    //不是质数
                    return false;
                }
            }
            return true;
        }
        return false;
    }
};
```

## 解题思路二：

滑动窗口，每次只保留第一个值和最后一个值，如果最后一个值=-1，说明只有一个或者0个质数，返回结果为0，如果有两个及以上质数，dp[1]的下标一定>-1，结果为dp[1]-dp[2]

```
class Solution {
public:
    int maximumPrimeDifference(vector<int>& nums) {
        //滑动窗口
        int dp[2] = {-1,-1};
        for (int i = 0; i < nums.size(); i++)
        {
            if (isrightVa(nums[i]))
            {
                if (dp[0] !=-1)
                {
                    dp[1] = i;
                }
                else
                    dp[0] = i;
            }
        }

        return dp[1]>-1?dp[1] - dp[0]:0;
```

```
    }
private:
    bool isrightVa(int n)
    {
        if (n == 1) return false;
        if (n > 1) {
            for (int i = 2; i <= sqrt(n); i++)
            {
                if (n % i == 0)
                {
                    //不是质数
                    return false;
                }
            }
            return true;
        }
        return false;
    };

  };
```

## 解题思路三：

双向遍历，使用两个整型记录正向和反向遍历的第一个质数的下标，最后这两个下标之差一定是最大质数距离

```
class Solution {
    bool is_prime(int n) {
        for (int i = 2; i * i <= n; i++) {
            if (n % i == 0) {
                return false;
            }
        }
        return n >= 2;
    }

public:
    int maximumPrimeDifference(vector<int>& nums) {
        int i = 0;
        //正向遍历
        while (!is_prime(nums[i])) {
            i++;
        }
        //反向遍历
        int j = nums.size() - 1;
        while (!is_prime(nums[j])) {
            j--;
        }
        return j - i;
    }
};
```