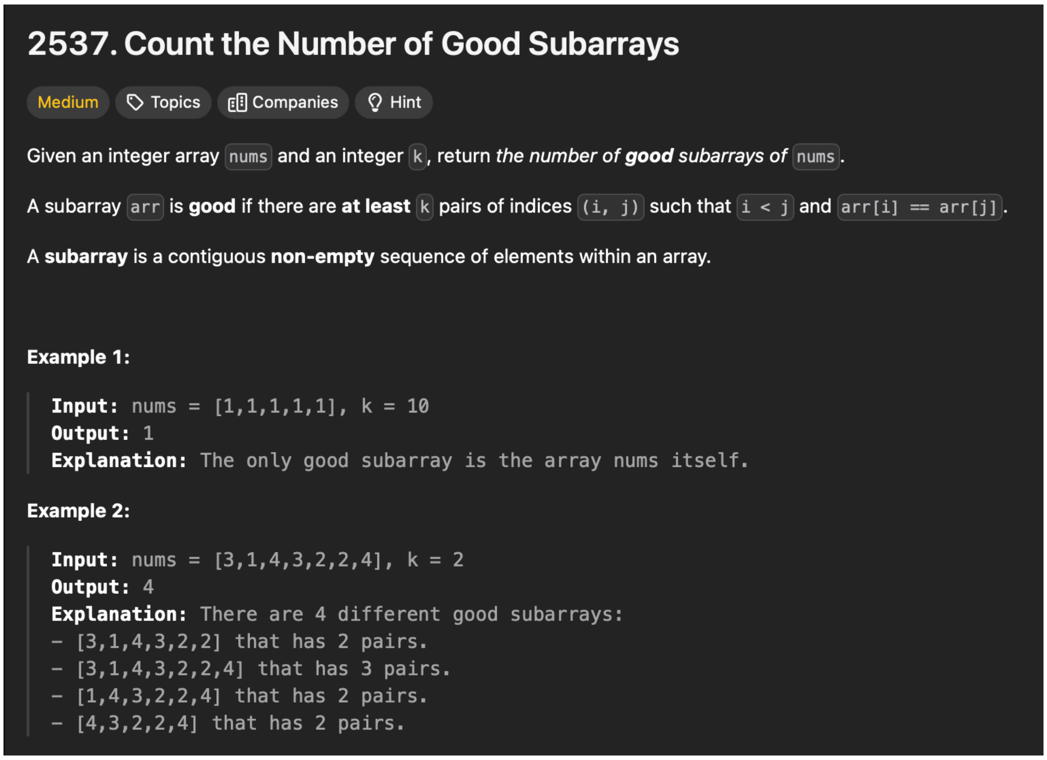


2537. Count the Number of Good Subarrays

Given an integer array `nums` and an integer `k`, return *the number of **good** subarrays of* `nums`.

A subarray `arr` is **good** if there are **at least** `k` pairs of indices `(i, j)` such that `i < j` and `arr[i] == arr[j]`.

A **subarray** is a contiguous **non-empty** sequence of elements within an array.

Example 1:

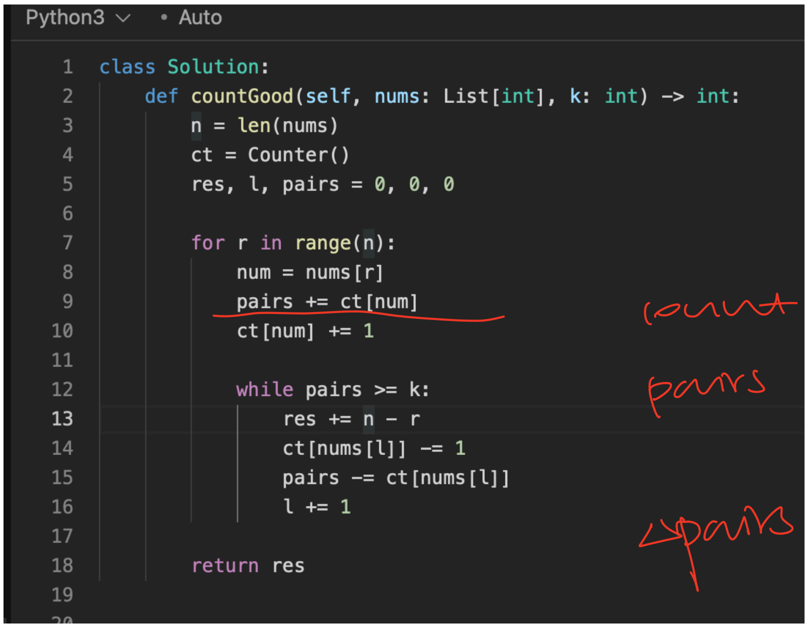
```
Input: nums = [1,1,1,1,1], k = 10
Output: 1
Explanation: The only good subarray is the array nums itself.
```

Example 2:

```
Input: nums = [3,1,4,3,2,2,4], k = 2
Output: 4
Explanation: There are 4 different good subarrays:
- [3,1,4,3,2,2] that has 2 pairs.
- [3,1,4,3,2,2,4] that has 3 pairs.
- [1,4,3,2,2,4] that has 2 pairs.
- [4,3,2,2,4] that has 2 pairs.
```

① Counter + Sliding window

方法想到，但是写不出来.



```python
class Solution:
    def countGood(self, nums: List[int], k: int) -> int:
        n = len(nums)
        ct = Counter()
        res, l, pairs = 0, 0, 0

        for r in range(n):
            num = nums[r]
            pairs += ct[num]
            ct[num] += 1

            while pairs >= k:
                res += n - r
                ct[nums[l]] -= 1
                pairs -= ct[nums[l]]
                l += 1

        return res
```

count $n \rightarrow n+1$

pairs $\frac{n(n-1)}{2} \rightarrow \frac{(n+1)n}{2}$

$\Delta pairs = \frac{(n+1)n}{2} - \frac{n(n-1)}{2} = n$