

```

class Solution:
    def numSubarrayProductLessThanK(self, nums: List[int], k: int) -> int:
        i, j = 0, 0
        product = 1
        subarrays = 0

        while j < len(nums):
            if j < len(nums) - 1 and product * nums[j] < k:
                product *= nums[j]
                j += 1
            elif i < j:
                subarrays += j - i
                product //= nums[i]
                i += 1
            else:
                i += 1
                j += 1

        return subarrays

```

[Explore](#)
[Problems](#)
[Interview](#)
[Contest](#)
[Discuss](#)
[Store](#)

00:00:00

Premium

Description
Solution
Video Solutions
Discuss (9...)
Submission...

Python3
Autocomplete

### 424. Longest Repeating Character Replacement

Medium 6960 269 Add to List Share

You are given a string `s` and an integer `k`. You can choose any character of the string and change it to any other uppercase English character. You can perform this operation at most `k` times.

Return the length of the longest substring containing the same letter you can get after performing the above operations.

**Example 1:**

**Input:** `s = "ABAB", k = 2`  
**Output:** 4  
**Explanation:** Replace the two 'A's with two 'B's or vice versa.

**Example 2:**

**Input:** `s = "AABABBA", k = 1`  
**Output:** 4  
**Explanation:** Replace the one 'A' in the middle with 'B' and form "AABBBBA". The substring "BBBB" has the longest repeating letters, which is 4.

**Constraints:**

- `1 <= s.length <= 105`

```

1 class Solution:
2     def characterReplacement(self, s: str, k: int) -> int:
3         max_length = 0
4         for i in range(ord('A'), ord('Z') + 1):
5             left = 0
6             ops = k
7             target = chr(i)
8
9             for right in range(len(s)):
10                if s[right] != target:
11                    ops -= 1
12                while ops <= 0 and s[right] != target:
13                    if s[left] != target:
14                        ops += 1
15                        left += 1
16                max_length = max(max_length, right - left + 1)
17
18            return max_length
19
20
21

```

NEW

Console
Contribute

Run Code
Submit

```
class Solution:
    def romanToInt(self, s: str) -> int:
        _dict = {
            'I' : 1,
            'V' : 5,
            'X' : 10,
            'L' : 50,
            'C' : 100,
            'D' : 500,
            'M' : 1000
        }

        subtract = {
            'I' : set(['V', 'X']),
            'X' : set(['L', 'C']),
            'C' : set(['D', 'M'])
        }

        ans = 0
        i = 0
        while i < (len(s)):
            ch = s[i]
            if ch in subtract and i < len(s) - 1 and \
                s[i + 1] in subtract[ch]:
                ans += (_dict[s[i + 1]] - _dict[ch])
            else:
                ans += _dict[ch]
            i += 1
        return ans
```