Problem #713: Subarray Product Less Than K (Medium)

<https://leetcode.com/problems/subarray-product-less-than-k/>

My Solution:

1. The runtime for my solution is O(n^2).

2. If k <= 1 then there are no possible numbers less than k since nums is a positive array.

3. If nums is an array of 1’s only, then number of subarray = 1 + 2 + 3 +…+ n = (n) \* (n+1)//2

4. For each number in the array, check the product with the remaining numbers to the right one at time until the product <= k. Once product reaches k or more, or we reach the last number in the array, we start with the next number and check the numbers to the right.

This solution gave me Time Limit Exceeded Error.

class Solution:

def numSubarrayProductLessThanK(self, nums: List[int], k: int) -> int:

if k <= 1:

return(0)

n = len(nums)

if max(nums) == 1:

return ((n)\*(n+1)//2)

count = 0

for i in range(n):

prod = 1

done = False

j = i

while j < n and not done:

prod \*= nums[j]

if prod < k:

count += 1

else: # Once prod is exceeded, no point checking further since nums is an array of positive integers.

done = True

j += 1

return count

---------------------------

Then I found this solution online which is very good.

Ref: <https://www.youtube.com/watch?time_continue=19&v=4775IgUKfww&feature=emb_title>

class Solution:

def numSubarrayProductLessThanK(self, nums: List[int], k: int) -> int:

left = 0

result = 0

product = 1

for right in range(len(nums)):

product \*= nums[right]

if product >= k:

while product >= k and left <= right:

product /= nums[left]

left += 1

result += right - left + 1

return(result)