Max Consecutive Ones

Given a binary array nums, return the maximum number of consecutive 1's in the array.

Input: nums = [1,1,0,1,1,1]

Output: 3

Explanation: The first two digits or the last three digits are consecutive 1s. The maximum number of consecutive 1s is 3.

# Solution:

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

    ans, c = 0, 0

    for i in nums:

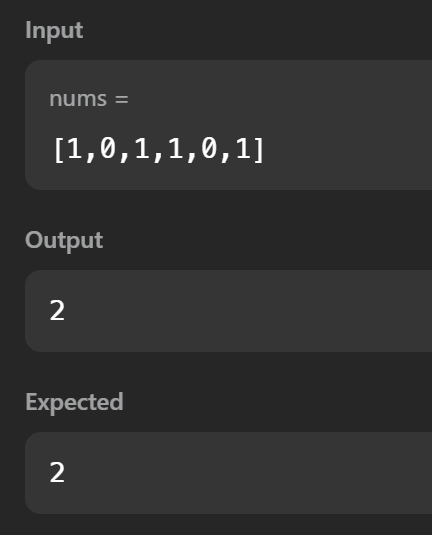
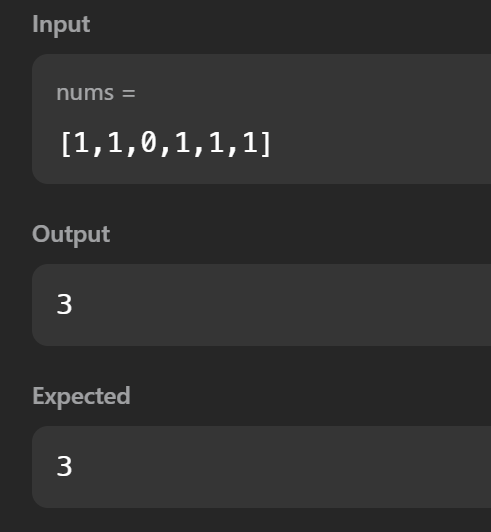
        if i==1: c += 1

        else: c = 0

        ans = max(c,ans)

    return ans

Output:



# Explanation:

There are 2 variables max and current. Current keeps track of the current count of consecutive ones. It is incremented when 1 is read and reset when 0 is read. In every iteration, the max value is set as maximum between max and current.

Time Complexity: O(n)

Space Complexity: O(1)

Longest Substring with At Least K Repeating Characters

# Given a string s and an integer k, return the length of the longest substring of s such that the frequency of each character in this substring is greater than or equal to k. If no such substring exists, return 0.

# Input: s = "aaabb", k = 3

Output: 3

Explanation: The longest substring is "aaa", as 'a' is repeated 3 times.

# Solution:

def longestSubstring(self, s: str, k: int) -> int:

    if len(s) < k:

        return 0

    char\_freq = {}

    for char in s:

        char\_freq[char] = char\_freq.get(char,0)+1

    for i, char in enumerate(s):

        if char\_freq[char] < k:

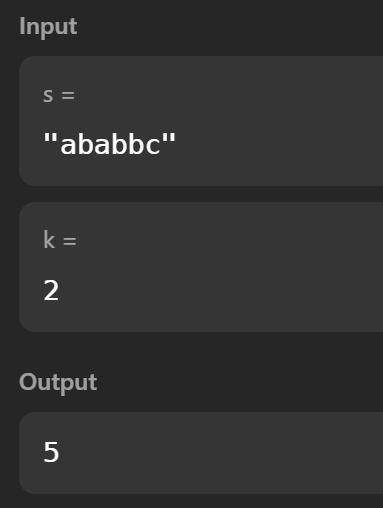
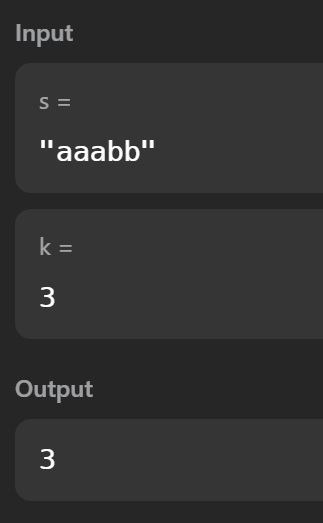
            left = self.longestSubstring(s[:i], k)

            right = self.longestSubstring(s[i+1:], k)

            return max(left, right)

    return len(s)

Output:

Time Complexity: O(n\*2)

Subarray Sum Equals K

Given an array of integers nums and an integer k, return the total number of subarrays whose sum equals to k. A subarray is a contiguous non-empty sequence of elements within an array.

Input: nums = [1,2,3], k = 3

Output: 2

# Solution:

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

    count = 0

    sum = 0

    dic = {}

    dic[0] = 1

    for i in range(len(nums)):

        sum += nums[i]

        if sum-k in dic:

            count += dic[sum-k]

        dic[sum] = dic.get(sum, 0)+1

    return count

Output:

