Problem # 413 : Arithmetic Slices (Medium)

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

<https://leetcode.com/problems/arithmetic-slices/>

My Solution:

<https://leetcode.com/problems/arithmetic-slices/discuss/1072534/Simple-Python-3-Solution-Runtime-beats-99.5-and-Memory-beats-100>

Runtime beats 99.50%, Memory usage beats 100%

1. Let n be the length of A.

2. If n is less than 3, then return 0 as we need at least 3 elements.

3. If n is 3, then check whether the difference between A[1] and A[0] is equal to the

Difference between A[2] and A[1]. If the differences are equal, return 1.

Otherwise, return 0.

4. Let d be the common difference which is the difference of the element at index 1 and element at index 0.

5. Initialize slice\_len\_list to an empty list.

6. Initialize count to 0 where count is used to track the number of pairs with the same common difference.

7. Iterate through the elements of list A from index 1 through index (n-1).

8. If the difference between the element of list A at index “i” and element at the previous index (i.e. element at index (i-1) is d, then increment count by 1. If not, append to slice\_len\_list the count of elements in the slice which are arithmetic. This will be one more than the number of consecutive pairs given by count. So (count + 1) is appended to slice\_len\_list.

9. If “I” is the index of the last element, then append (count + 1) to slice\_len\_list.

10. Now, calculate the number of slices. For this, set result to 0. Note that result is the variable that will hold the number of slices.

11. Iterate through the numbers in slice\_len\_list. If the number is 3, then increment the result by 1 since we need at least 3 numbers in an arithmetic slice. If the number is more than 3, then increment result by (num - 1) \* (num - 2)//2.

Explanation for using (num - 1) \* (num - 2)//2 :

If the number is 4, we have a list say [1, 2, 3, 4], we have 2 slices of length 3 and 1 slice of length 4.

If the number is 5, we have a list say [1, 2, 3, 4, 5], we have 3 slices of length 3 (i.e. [1,2,3], [2,3,4], [3,4,5]), we have 2 slices of length 4 (i.e. [1,2,3,4], and [2,3,4,5]), and we have 1 slice of length 5 which is [1,2,3,4,5].

So, if number in slice\_len\_list is k, number of slices we have will be ((k-2) + (k-1) +….+ 1). This is an Arithmetic series, whose sum = count of numbers \* average of first and last values.

Therefore, sum of (1 + 2 + …. + k-2) = (k-2) \* (1 + k -2) // 2.

That is, sum of (1 + 2+ … + k-2) = (k-2) \* (k-1)//2

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

class Solution:

def numberOfArithmeticSlices(self, A: List[int]) -> int:

n = len(A)

if n < 3:

return 0

elif n == 3: # if there are only 3 numbers in the array

if (A[1] - A[0]) == (A[2] - A[1]): # check the common difference

return 1

else:

return 0

d = A[1] - A[0] # common difference

slice\_len\_list = []

count = 0 # count of pairs with the difference of d

for i in range(1, n):

if A[i] - A[i-1] == d:

count += 1

else:

slice\_len\_list.append(count + 1) # save count of slice before resetting it

d = A[i] - A[i-1] # reset d to the new difference

count = 1 # reset count to 1

if i == n - 1:

slice\_len\_list.append(count + 1)

# Calculate number of slices

result = 0

for num in slice\_len\_list:

if num == 3:

result += 1

elif num > 3:

result += (num - 1) \* (num - 2)//2

return result