Problem # 1010 : Pairs of Songs with total duration Divisible by 60 (Medium)

<https://leetcode.com/problems/pairs-of-songs-with-total-durations-divisible-by-60/>

My Solution :

<https://leetcode.com/problems/pairs-of-songs-with-total-durations-divisible-by-60/discuss/965445/Simple-Python-3-Solution-Runtime-beats-99.86>

1. Create a list called alist where we take the modulo 60 of each element in time array.

2. Find the frequency of elements in alist using Counter from collections to create a dictionary called time\_dict.

3. Initialize count to 0.

4. Iterate through value of i from 0 through 31 using range,

5. Let n be the value of time\_dict with index I.

6. If i is 0 or 30, we need more than one such value to create combination pairs.

The number of combination pairs will be nC2 = n \* (n - 1) // 2.

Increment count by the number of combination pairs.

7. If i is not 0 and is not 30 (this is applying De Morgan’s Law by which

(A U B)c = Ac ∩ Bc

where union represents “or” and intersection represents “and”).

take the complement of i which is (60 – i). (Note that the sum of i and its complement is 60).

If the complement is more than 1, then number of possible pairs = (frequency of i) times ( frequency of the complement).

Increment count by the number of combination pairs.

NOTE: The complement of a number from 1 through 29, will be in the range 31 through 59.

8. Return the value of count.

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

from collections import Counter

class Solution:

def numPairsDivisibleBy60(self, time: List[int]) -> int:

alist = [ x % 60 for x in time]

time\_dict = Counter(alist)

count = 0

for i in range(31):

n = time\_dict[i]

if (i == 0 or i == 30) and n > 1:

count += n \* (n - 1)//2 # nC2 = n \* (n - 1) // 2

elif (i != 0 and i != 30) and (60 - i) in time\_dict:

count += n \* time\_dict[60 - i]

return count