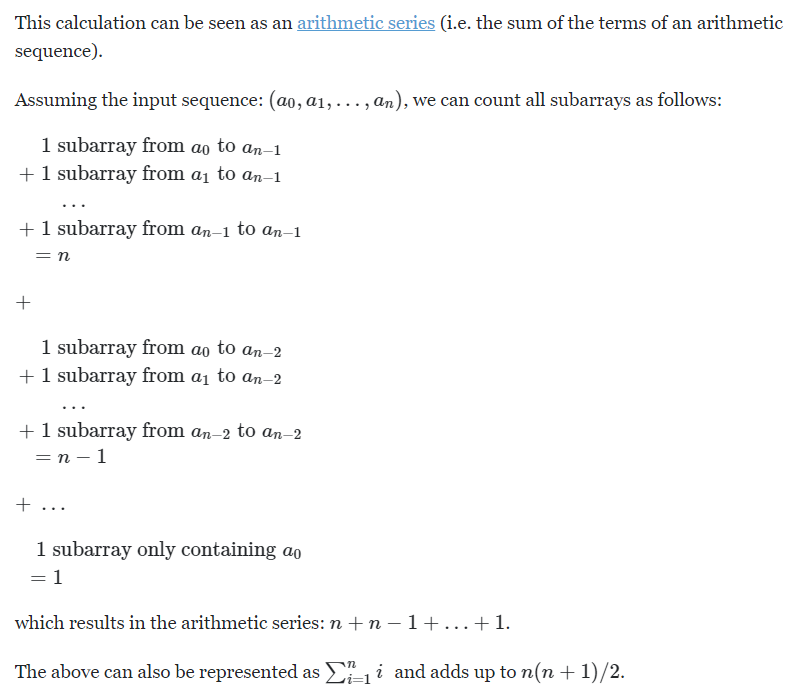
The concept to find total number of contiguous sub-arrays in an array of size n: **(n\*(n+1))/2**

**Proof:**



Example:

**1 3 5 7**

{1} {1 3} {1 3 5} {1 3 5 7} - 4 //Taking Every Combination With 1

{3} {3 5} {3 5 7} - 3

{5} {5 7} - 2

{7} - 1

TOTAL: 4 + 3 + 2 + 1 = 10

OR (REVERSE)

{1 3 5 7} {3 5 7} {5 7} {7} - 4 //Taking Every Combination With 7

{1 3 5} {3 5} {5} - 3

{1 3} {3} - 2

{1} - 1

TOTAL: 4 + 3 + 2 + 1 = 10

If the window of size (j-i+1) is following the condition: The product of all the elements in the window of size n is < maxproduct, then all the subarrays of the subarray (window) is the answer.

How to calculate without overlapping problem:

Taking an example: {10,5,2,6} & MaxAllowed Product: 101

First Window: i=0 j=0

|10| 5 2 6 total = 0 - 0 + 1 = 1

{10} Taking every combination with 10.

Second Window: i=0 j=1

|10 5| 2 6 total = 1 - 0 + 1 = 2

{10, 5} {5} Taking every combination with 5.

Third Window: i=0 j=2

|10 5 2| 6 total = 2 - 0 + 1 = 3

{10, 5, 2} {5, 2} {2} Taking every combination with 2.

Fourth Window: i=0 j=3

|10 5 2 6| Product = 600 > 101

while(product > 101) shrink the window:

Product = product/array[i]; i++;

New Window: i=1 j=3

10 |5 2 6| total = 3 - 1 + 1 = 3

{5, 2, 6} {2, 6} {6} Taking every combination with 6.

TOTAL : 1 + 2 + 3 + 3 = 9

There are a total of 9 subarrays in the array {10 5 2 6}, such that their product is less than 101.