Q→ Given a char array of lower case characters. (a – z)
Count the # pairs (i, j) s.t. i < j,
A[i] = 'a' & A[j] = 'g'.

$A = [\overset{0}{a}\ \overset{1}{b}\ \overset{2}{e}\ \overset{3}{g}\ \overset{4}{a}\ \overset{5}{g}]$

i	j
0	3
0	5
4	5

Ans = 3

$A = [\overset{0}{a}\ \overset{1}{c}\ \overset{2}{g}\ \overset{3}{d}\ \overset{4}{g}\ \overset{5}{a}\ \overset{6}{g}]$

i	j
0	2
0	4
0	6
5	6

Ans = 4

$A = [\overset{0}{b}\ \overset{1}{c}\ \overset{2}{a}\ \overset{3}{g}\ \overset{4}{g}\ \overset{5}{a}\ \overset{6}{a}\ \overset{7}{g}]$

i	j
2	3
2	4
2	7
5	7
6	7

Ans = 5

Bruteforce →

```
cnt = 0
for i → 0 to (N-2) {
    if ( A[i] == 'a')
        for j → (i+1) to (N-1) {
            if ( A[j] == 'g')
                cnt ++
        }
}
return cnt
```

→ count #'g' from (i+1) to (N-1)

(i+1) ⟶ end

TC = O(N²)

SC = O(1)

←

	0	1	2	3	4	5	6
A =	[a	c	g	d	g	a	g]
cnt #'g' →	3	3	3	2	2	1	1

(i+1) |——→ end

Prefix Sum done L←R is <u>Suffix Sum</u>.

```
if(A[N-1] == 'g')   cntg[N-1] = 1
else   cntg[N-1] = 0
for i → (N-2) to 0 {
        if ( A[i] == 'g')
                cntg[i] = 1 + cntg[i+1]
        else
                cntg[i] = cntg[i+1]
}
```

calculate

TC = <u>O(N)</u>

SC = <u>O(N)</u>

	0	1	2	3	4
A =	[g	b	e	g	a]
cntg =	[2	1	1	1	0]

cntg[i] → #'g' from index i to (N-1).

```
ans = 0
for i → 0 to (N-2) {
        if ( A[i] == 'a')
                ans += cntg[i+1]
}
return ans
```

use

TC = <u>O(N)</u>

Total   TC = O(N + N) = <u>O(N)</u> ✓   SC = <u>O(N)</u> → can it reduce?

<u>calculate & use together</u>

ans = 0

```
if(A[N-1] == 'g')   cnt = 1
else   cnt = 0
for i → (N-2) to 0 {
   → if ( A[i] == 'g')
          cnt = 1 + cnt
   → if ( A[i] == 'a')
          ans += cnt
} return ans
```

$$\begin{array}{cccccccc} & & & & i & & & \\ & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ A = [& a & c & g & d & g & a & g] \end{array}$$

cnt = $\cancel{1}\ \cancel{2}\ 3$

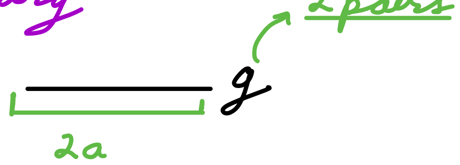
ans = 0 + 1 + 3 = 4

calculate & use → carry forward

TC = O(N)   SC = O(1)

H.W → Try the same task by calculating # 'a' from index 0 to (i-1).

0 ← (i-1)



ans = Σ count of 'a' on left of index i
∀i, A[i] = 'g'

Subarray → continuous part of array.

A = [2 4 1 6 -3 7 8 4]

single element ✓

complete array ✓

start ——— end

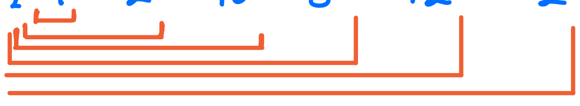
OR

start & length

$$\begin{array}{cccc} & 0 & 1 & 2 \\ A = [& 1 & 2 & \textcircled{3}] \end{array}$$

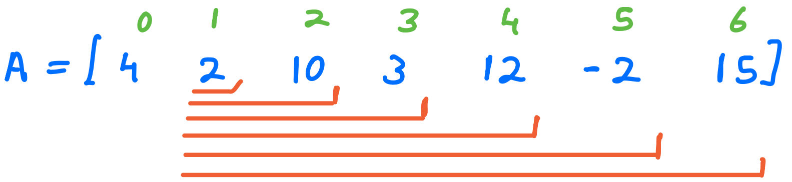
subarray that start at i=2 → 1

$$\begin{array}{cccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ A = [& 4 & 2 & 10 & 3 & 12 & -2 & 15] \end{array}$$



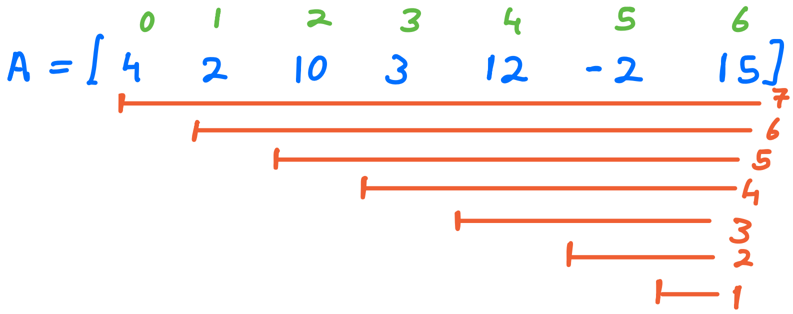
subarray starting

from index 0 = 7



\# subarray start from index 1 = 6

Total # subarrays of array size N →



start	#subarray
0	7
1	6
2	5
3	4
4	3
5	2
6	1

$$\underset{N}{\boxed{7}} + 6 + 5 \ldots + 1$$

$$N + (N-1) + (N-2) + \ldots + 1 = \frac{N*(N+1)}{2}$$

Q → Print all subarrays of the array.

10:40 PM

$$A = \begin{array}{ccc} 0 & 1 & 2 \\ [1 & 2 & 3] \end{array}$$

1

1 2

1 2 3

2

2 3

3

```
for i → 0 to (N-1) {
    for j → i to (N-1) {
        for k → i to j {
            print (A[k])
        }
        print ("\n")
    }
}
```

st   end
i —— j

TC = $\underline{O(N^3)}$

SC = $\underline{O(1)}$

Q → Given an array of integers, find the length of smallest subarray which contains both max & min element of array.

$\quad\ \ 0\ \ 1\ \ 2\ \ 3\ \ 4\ \ 5$

$A = [3\ \boxed{6}\ 2\ 1\ \boxed{6}\ 5]$ Ans = $\underline{2}$

$\quad\ \ 0\ \ 1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 6\ \ 7\ \ 8\ \ 9\ \ 10$

$A = [2\ 2\ 6\ 4\ 5\ 1\ 5\ 2\ 6\ 4\ 1]$

min = 1

max = 6   Ans = $\underline{3}$

Bruteforce → ∀ subarrays, iterate & check. TC = $\underline{O(N^3)}$

SC = $\underline{O(1)}$

smallest subarray   min —————— max

max —————— min

max   max   min

min ———————— max

∀ max element, find closest min element on left.

[L R] → R-L+1

0 1 2 3 4 5 6 7 8 9 10
A = [2 2 6 4 5 1 5 2 6 4 1]
i

min = 1   idx_min = ~~-1~~ ~~5~~ 10   length → [5 8]
max = 6   ans = 4   = 8 - 5 + 1 = 4

max ———————— min

0 1 2 3 4 5 6 7 8 9 10
A = [2 2 6 4 5 1 5 2 6 4 1]
i

min = 1   idx_max = ~~2~~ 8   [2 5] → 5 - 2 + 1 = 4
max = 6   ans = ~~4~~ 3   [8 10] → 10 - 8 + 1 = 3

```
//calculate minA & maxA
idx_max = -1
idx_min = -1            ans = N  //worst case answer
for i → 0 to (N-1) {
    if (A[i] == minA) {
        if (idx_max != -1)        max |———— i
            ans = min (ans, i-idx_max +1)
        idx_min = i
    } else if (A[i] == maxA) {
        if (idx_min != -1)        min |———— i
            ans = min (ans, i-idx_min+1)
        idx_max = i
```

```
    }
}
```

TC = O(N)   SC = O(1)