Q1 Given a string..

↓

collection / array

of characters

⇓

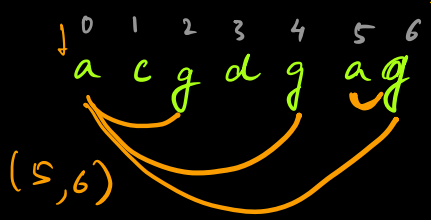
"ankit" ⇒ ['a', 'n', 'k', 'i', 't']

Find no. of pairs (i, j) s.t.

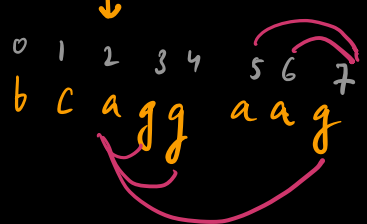
$i < j$

and $a[i] = 'a'$ $a[j] = 'g'$

Example:



(0,2) (5,6)
(0,4)
(0,6) ⇒ (4) pairs



(5) ✓

a a a a a g

! !

cnt = 0

for i in range (len(a)):

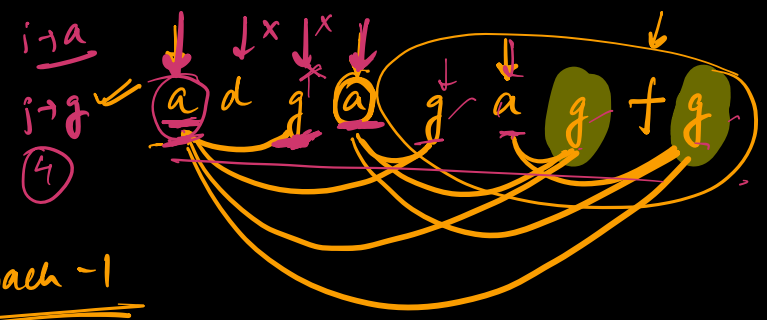
⇒ for j in range (i+1, len(a)):

if a[i] == 'a' and a[j] == 'g':

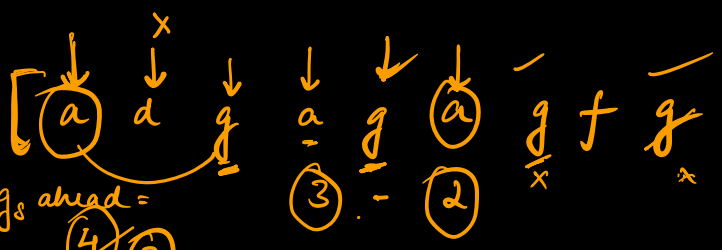
cnt += 1

$O(N^2)$

SC = ? $O(1)$



Approach - 1



O(N) operation

gs ahead = 4 + 3 + 2 = 9

'ag'

Optimised BF

g a d g a g a g f g

(1) Count no. of gs in the string

=> no. of g = 4

=> no. of gs ahead

ans = 4

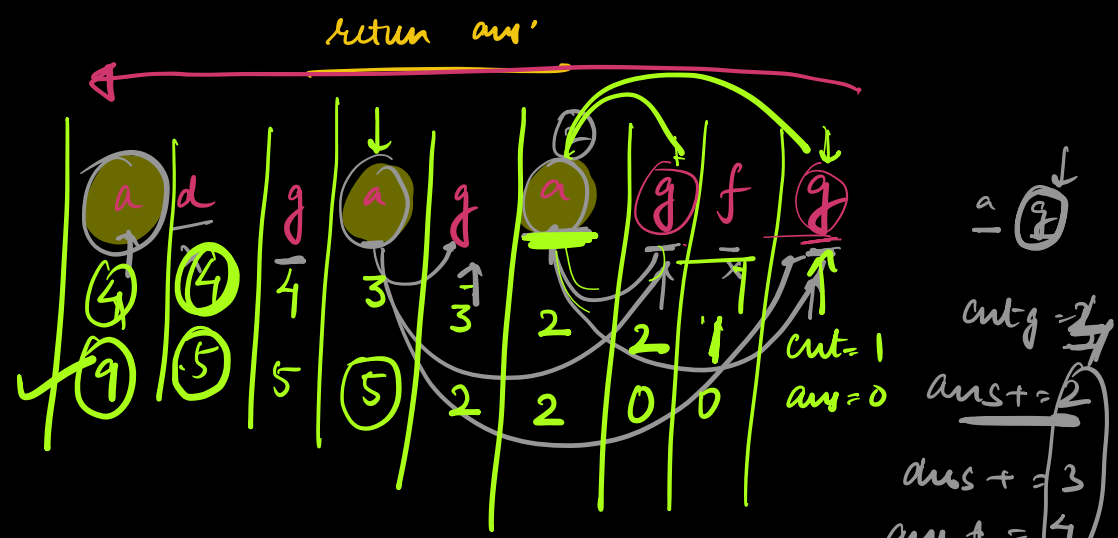
```
cnt_g = 0
for i in s:
    if i == 'g':
        cnt_g += 1
```

(1) → O(N)

SC(O(1))

```
ans = 0
for i in s:
    if i == 'a':
        ans += cnt_g
    elif i == 'g':
        cnt_g -= 1
```

(2) → O(N)



CFA

(N-1, -1, -1)

```
ans = 0   cnt = 0
for i in range (len(a)-1, -1, -1):
    if a[i] == 'g':
        cnt += 1
    if a[i] == 'a':
        ans += cnt

return ans
```

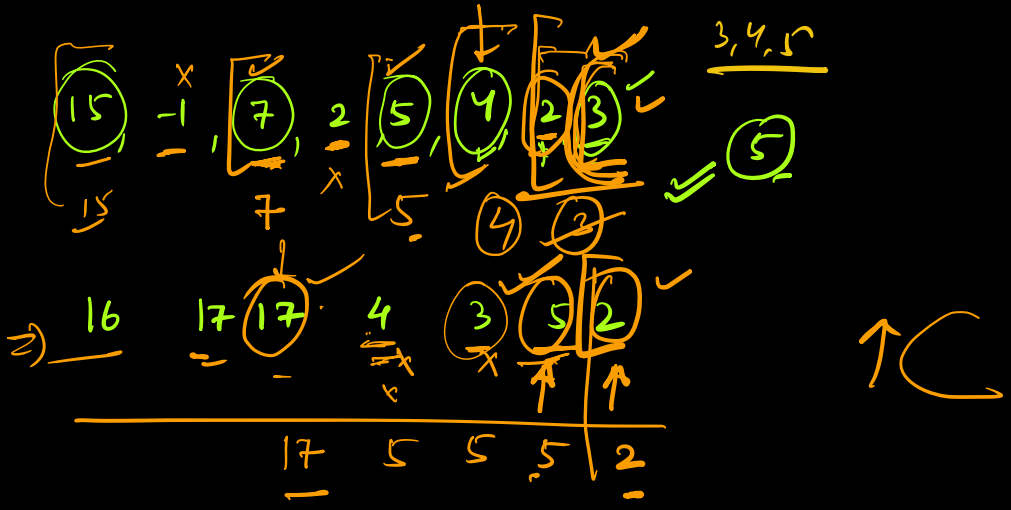
Q2 Leaders in Array

Given an array of N elements.
Count no. of leaders in array.

An element can be called a leader if it is strictly greater than all the elements on the right side

Note:
Last element is always a leader.

Example:



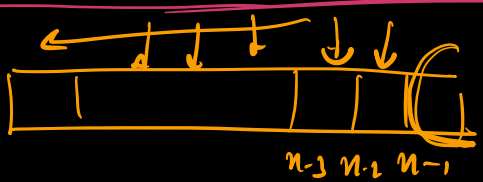
max

element > max

elements is bigger than all values

Leader

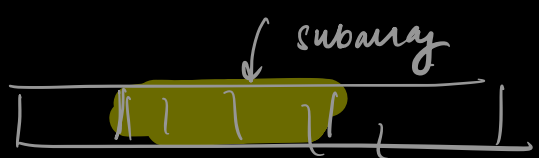
```
ans = 1
max = A[-1]   or A[len(A)-1]
for i in range(len(A)-2, -1, -1):
    if A[i] > max:
        ans += 1
        max = A[i]
return ans
```



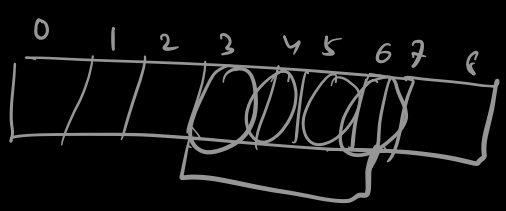
Subarrays

⇓

Continuous part of an array



index 3 to index 7



3 4 5 1 2 1

5 1 2 ⇒ a subarray

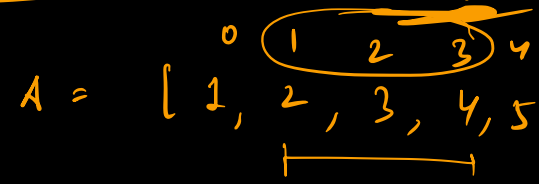
1 1 → a subarray

2 →

[ ] ⇒ not a subarray

⇓

a[i : j] ⇒ slicing



A[1 : 4] =

2, 3, 4

2, 3, 4, 5

(l)

O(N) → max(l) → max element of list

O(N) → min(l) → min

O(NlogN) l.sort() → sort.

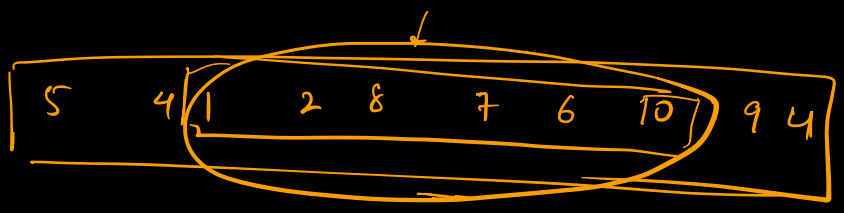
Break → 5 mins

(3) Closest Min Max

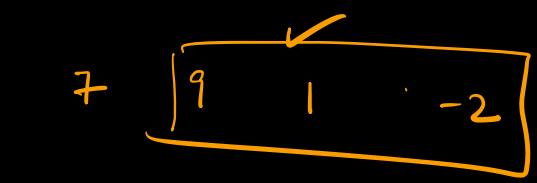
Given an array of N elements.
Find the length of smallest subarray which contains both min and max of the array.

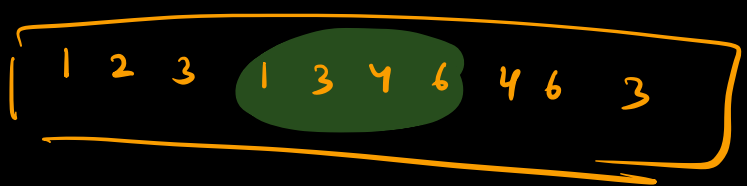
Obs - 1

Max and min should be at the boundary of subarray



(6)

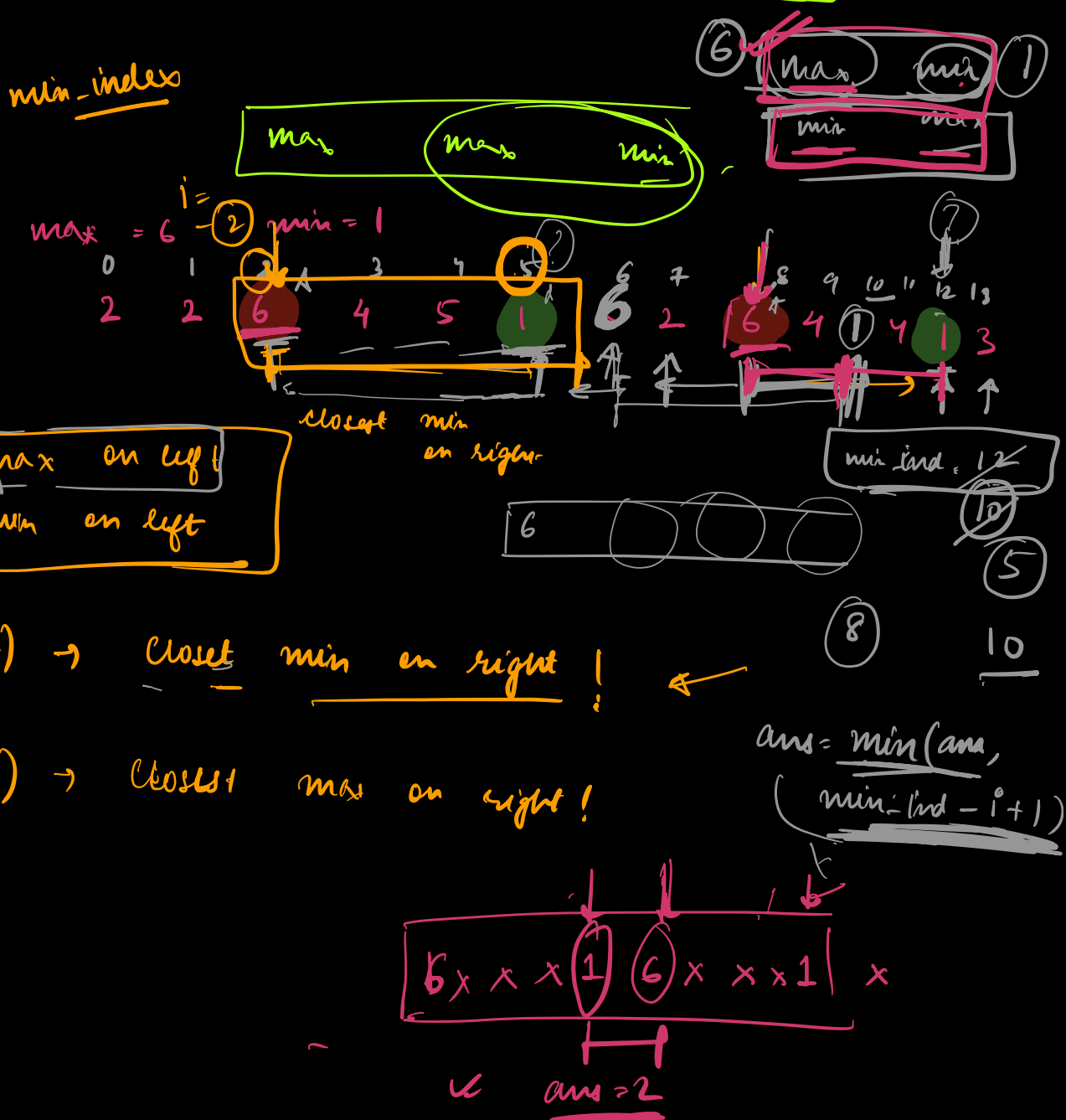




Observations :

① Max and min should be at boundary of the subarray

② Can there be more than 1 max / min in the min length subarray →



a) → Closest min on right !

b) → Closest max on right !

ans = min(ans, (min_ind − i + 1))

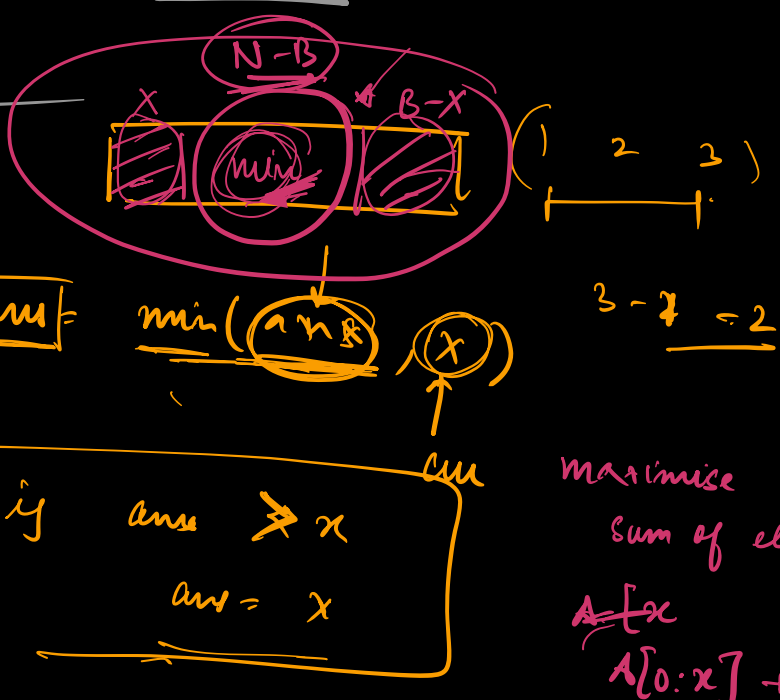
ans = len(l)
min_index = -1

min_value = min(l)
max_val = max(l)

for i in range(len(l)-1, -1, -1):
    if l[i] == max and min_index != -1:
        ans = min(ans, min_index - i + 1)
    if l[i] == min
        min_index = i

max_index = -1

max ↔ min



if ans > x
    ans = x

maximise sum of elements

A[x]

A[0:x] + A[x:n-1]

for $i \rightarrow (0 \rightarrow n)$