Carry Forward →

" "

Count pairs "ag"

Given a string, calculate the no. of pairs (i, j) such that $i < j$ && $s[i] == 'a'$ && $s[j] == 'g'$.

0	1	2	3	4	5	6	7
b	a	a	g	d	c	a	g
		i					j

$(1,3)$, $(1,7)$
$(2,3)$, $(2,7)$
$(6,7)$

ANS → 5

I) BF

```
cnt = 0;
for(i = 0; i < N; i++) {
    for(j = i+1; j < N; j++) {
        if(A[i] == 'a' && A[j] == 'g') {
            cnt++;
        }
    }
}
ret cnt;
```

TC = $O(N^2)$

SC = $O(1)$

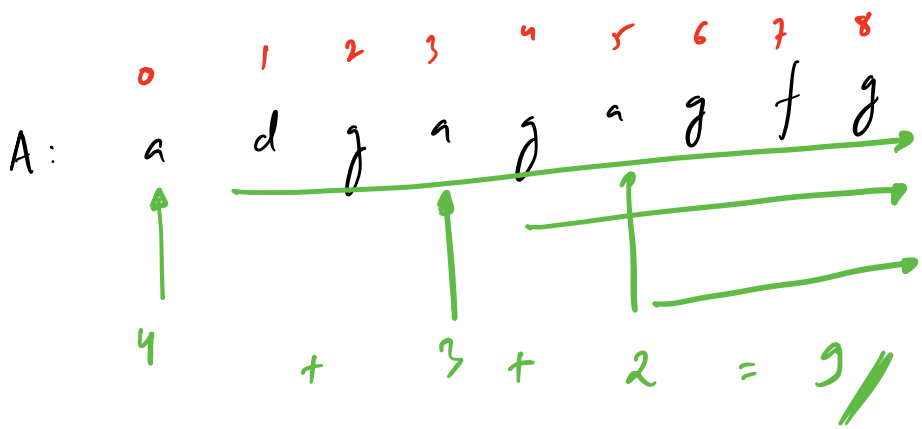
```
cnt=0;
for(i=0; i<N; i++) {
    if(A[i]=='a') {
        for(j=i+1; j<N; j++) {
            if(A[j]=='g'){
                cnt++;
            }
        }
    }
}
```

TC: $O(N^2)$

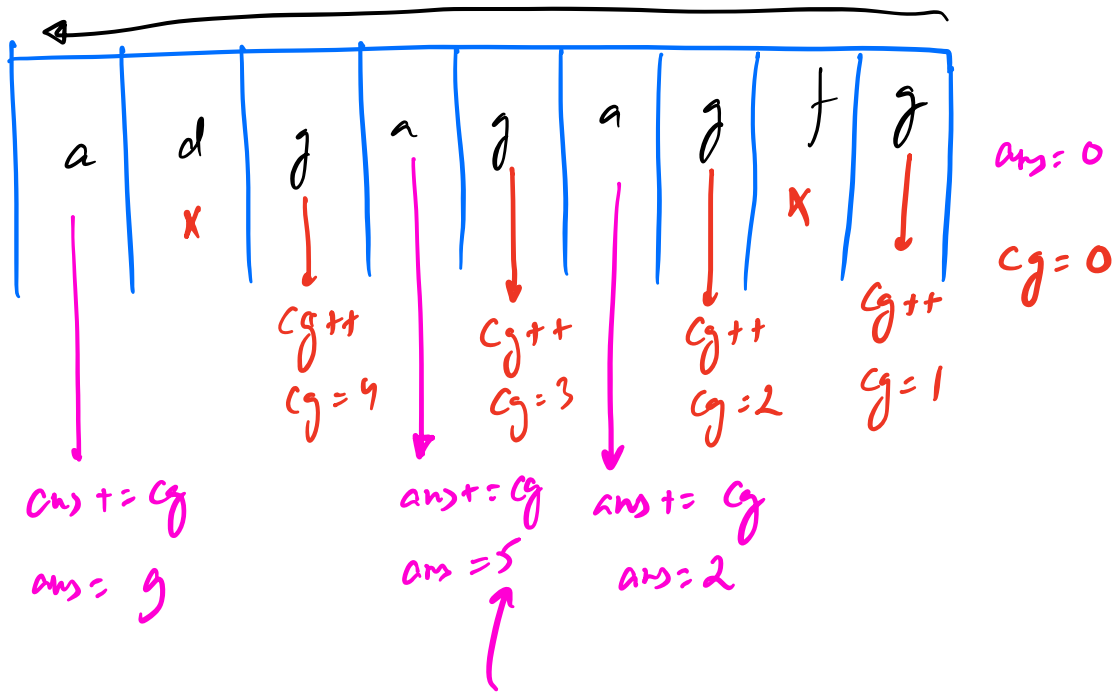
Worst Case:

a a a a — — a

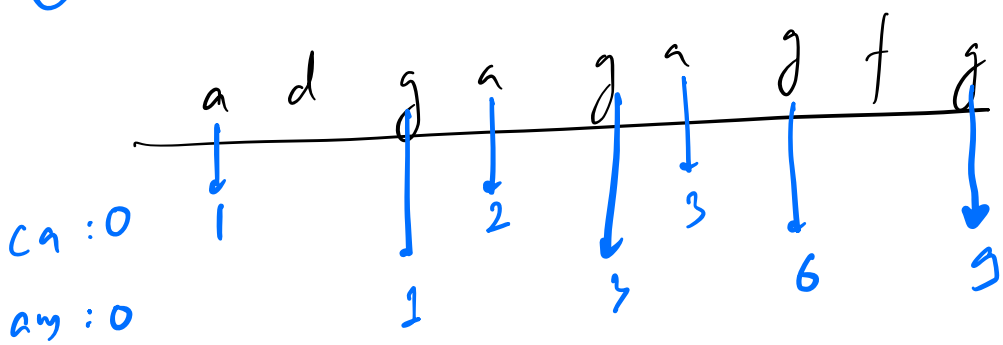
II



Obs: for every 'a', count the no. of 'g' on the right of it!



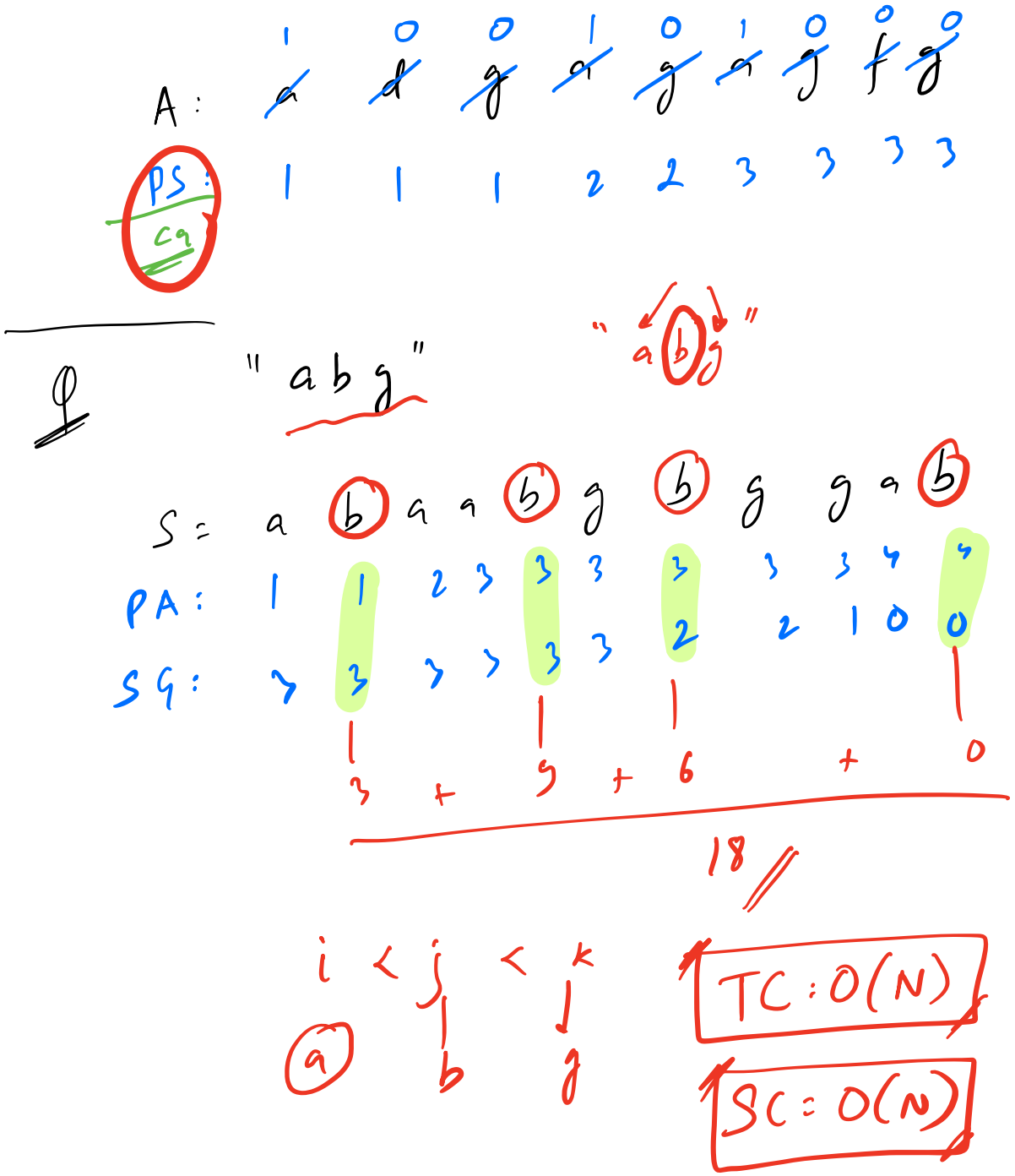
(b)



```
cg = 0, ans = 0;
for(i = N-1; i >= 0; i--) {
    if( A[i] == 'g') {
        cg++;
    }
    else if( A[i] == 'a') {
        ans += cg;
    }
}
ret ans;
```

TC : O(N)

SC : O(1)

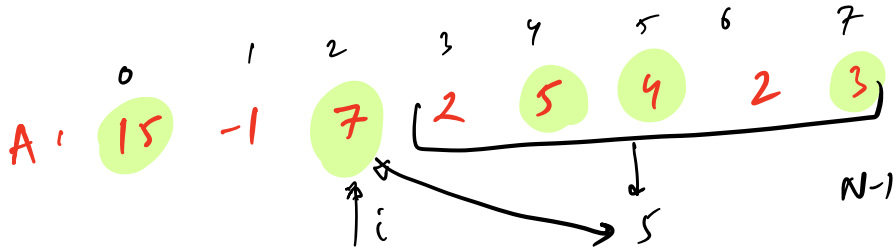

A:
PS:
cq
"abg"
"abj"
S: a b a a b g b g g a b
PA: 1 1 2 3 3 3 3 3 3 4 4
2 1 0 0
SQ:
3 + 3 + 6 + 0
18
i < j < k
a b g
TC: O(N)
SC: O(N)

Q Given an array A.

Find the no. of leaders in the array!

An element is a leader if it is greater than all elements on it's right!

NOTE: A[N-1] ↳ ALWAYS A LEADER



ANS: 5

I) BF

TC: $O(N^2)$

SC: $O(1)$

```
cnt=0
for(i=0; i<N; i++){
    maxi = -∞
    for(j=i+1; j<n; j++){
        maxi = max(maxi, A[j]);
    }
    if(A[i] > maxi){
        cnt++;
    }
}
```

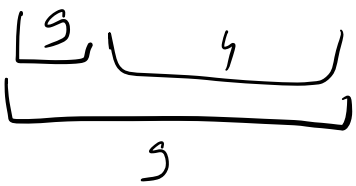
A: [10 7 9 3 2 4]
0 1 2 3 4 5

10 ← 9 ← 9 ← 4 ← 4 ← 4 ← mx = -∞

```
mx = -∞;
cnt = 0;
for( i = N-1; i >= 0; i--) {
    if( A[i] > mx) {
        cnt++;
        mx = A[i];
    }
}
ret cnt;
```

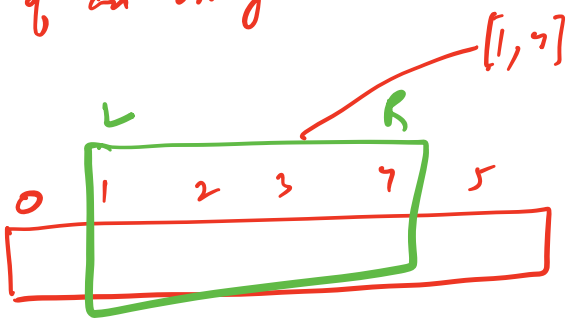
TC: O(N)

SC: O(1)



Sub Array →

→ Continuous portion of an array



SA → [L, R]

L <= R

elements ⇒ R - L + 1

[4, 4)

Closest Min Max.

Given an array, find the length of the smallest subarray which contains both the Min & MAX of the array!

	0	1	2	3	4	5	6	7	8	9
A:	1	2	3	1	3	4	6	4	6	3

[3, 6]

ANS = 4

MIN → 1

MAX → 6

1) BF

STEPS:

1. find MIN, MAX → O(N)
2. Consider every S.A → N^2
   - → check if that SA contains MIN & MAX → O(N)

N^2 × O(N) = N^3

→ YES : Consider

→ NO : ignore

TC : $O(N^3)$

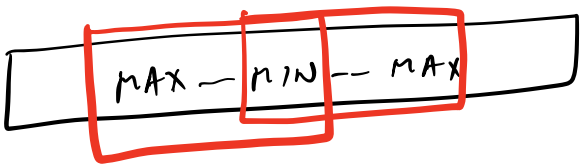
Eg: 8 8 8 8 8

ANS → 1

	0	1	2	3	4	5	6	7	8	9	10
A:	2	2	6	4	5	1	5	2	6	4	1

Obs :

1.

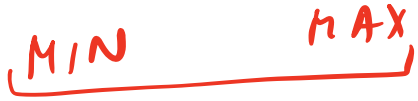


the ANS S.A. contain EXACTLY 1 occ of MAX, 1 occ. of MIN

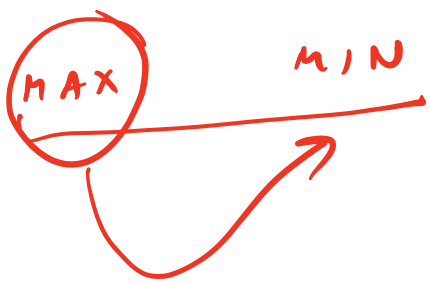
2.

→ In the ANS S.A. MAX & MIN would be on the border!

- CASE 1:



- CASE 2:

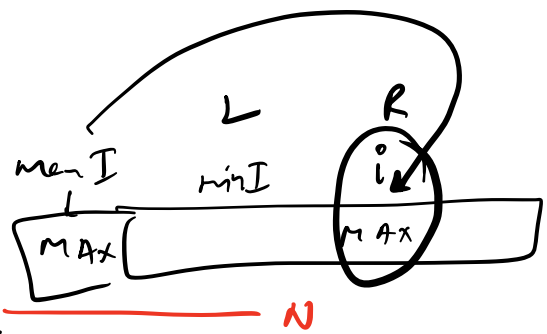


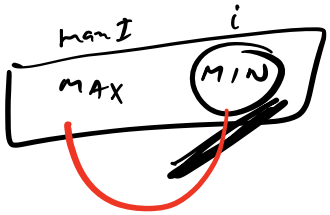
1. find MAX of the A → N
2. ——— MIN ——— → N

```
if (MIN == MAX)
    ret 1

ans = ∞
minI = maxI = -1;
for (i=0; i<N; i++) {
    if (A[i] == MAX) {
        if (minI != -1) {
            ans = min(ans, i - minI + 1);
        }
        maxI = i;
    }
    else if (A[i] == MIN) {
        if (maxI != -1) {
            ans = min(ans, i - maxI + 1);
        }
        minI = i;
    }
```

3.



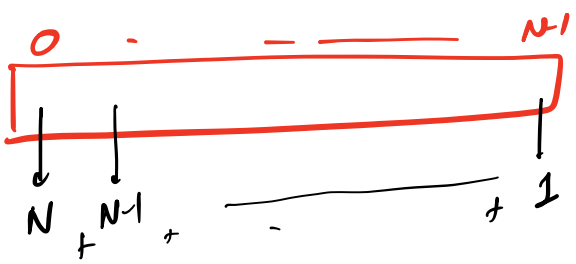


}

ret ans;

TC : O(N)

SC = O(1)



Total # SA = $\frac{N(N+1)}{2}$ → $O(N^2)$