$$ps[i] = ps[i-1] + A[i]$$

Q.1 Given an Array of size N, Build the array leftMax, s.t leftMax[i] ⇒ Max value in the array from 0 to i

	0	1	2	3	4	5	6	7	8	9
A:	-3	6	2	4	5	2	8	-9	3	1
leftMax[]:	-3	6	6	6	6	6	8	8	8	8

LM[0] = A[0]

LM[1] = max[0,1] = max(LM[0], A[1])

LM[2] = max[0,2] = max(LM[1], A[2])

LM[3] = max[0,3] = max(LM[2], A[3])

LM[6] = max[0,6] = max(LM[5], A[6])

leftMax[i] = max[0,i]

leftMax[i] = max(LM[i-1], A[i])

```
int LM[N];
LM[0] = A[0]
for(i = 1; i < N; i++){
    LM[i] = max(LM[i-1], A[i]);
}
```

TC: O(N)
SC: O(N)

Q.2 Given an Array of size N, Build the array RightMax, s.t RightMax[i] ⇒ Max value in the array from i to N-1

	0	1	2	3	4	5	6	7	8	9
A:	-3	6	2	4	5	2	8	-9	3	1
RightMax[]:	8	8	8	8	8	8	8	3	3	1

N-2 ← N-1

RM[N-1] = A[N-1]
RM[N-2] = max[N-2, N-1] = 3
RM[N-3] = max[N-3, N-2]
= max(RM[N-2], A[i])
RM[6] = max[6, 9]
= max(RM[7], A[6]) = 8

$RM[i] = max[i, N-1]$

$RM[i] = max(RM[i+1], A[i])$

```
int RM[N];
RM[N-1] = A[N-1];
for(i = N-2; i>=0; i--){
        RM[i] = max(RM[i+1], A[i]);
}
```

TC: O(N)
SC: O(N)

Q.3 Amazon Given a string of lowercase alphabets. Return the count of pairs (i,j) s.t
i < j
S[i] = 'a' } 'ag'
S[j] = 'g' }

Str: "a b e g a g"
0 1 2 3 4 5

(0,3) → 'ag'
(0,5) → 'ag'
(4,5) → 'ag'
} ⇒ 3

Quiz

S: a c g d g a g
0 1 2 3 4 5 6

(0,2) (0,4) (0,6) (5,6) ⇒ 4

Quiz

S: b c a g g a a g
0 1 2 3 4 5 6 7

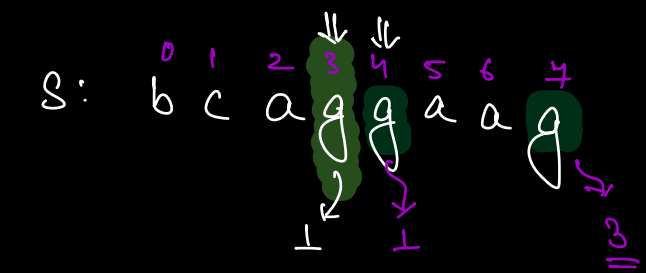
(2,3) (5,7) (6,7)
(2,4)
(2,7)
⇒ 5

⇒ Brute Force :- For every 'a' find the no. of g's on the right side.

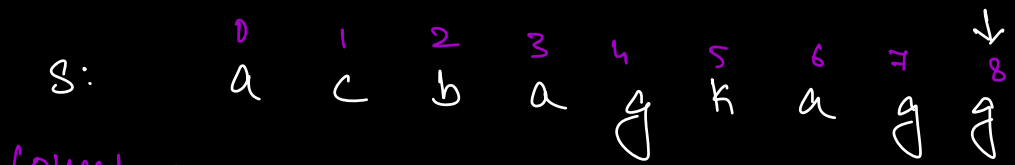
```
ans = 0
for( i = 0; i < N; i++ ){
    if( s[i] == 'a' ){
        for(j = i+1; j < N; j++ ){
            if( s[j] == 'g' ){
                ans++
```

3 3 3 3

TC: O(N²)
SC: O(1)

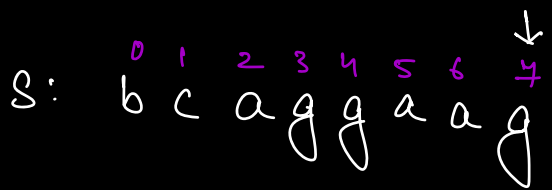


⇒ Every 'g' will make a valid pair with all the a's on the left side of it.



count_a = ~~0~~ ✗ ✗ 3

ans = ~~0~~ ✗ ✗ 8



count_a = ∅ ✗ ✗ 3

ans = ∅ ✗ ✗ 5

```
ans = 0
count = 0
for(i = 0; i < N; i++) {
        if(S[i] == 'a')
            count++
        else if( S[i] == 'g')
            ans += count;
}
return ans;
```

TC: O(N)
SC: O(1)

Q.4 Amazon

Given an Array of size N, return the length of smallest subarray which contains both min & max of the Array.

Contigous part of array.

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

Amax = 6

Amin = 1

[3, 6] ⇒ [i, j] ⇒ 6 - 3 + 1

= 4

Quiz

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

$A_{max} = 6$

$A_{min} = 1$

$[8, 10] \Rightarrow 10 - 8 + 1$

$\Rightarrow 3.$

Quiz

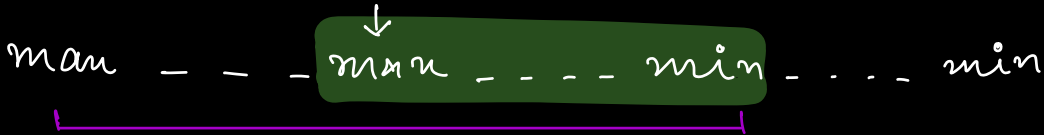
	0	1	2	3	4	5	6	7	8	9	10	11
A:	1	6	4	2	7	7	5	1	3	1	1	5

$A_{max} = 7$

$A_{min} = 1$

$[5, 7] \Rightarrow 7 - 5 + 1$

$= 8$



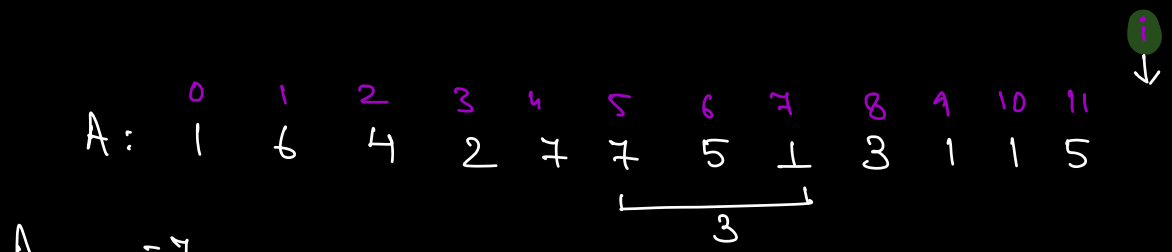
⇒ In the answer subarray, MAX & MIN will be corner of the subarray.

⇒ Can there be 2 max or 2 min in the ans subarray ? ⇒ NO.

⇒ There will be only ONE min & ONE max in the ans subarray.

A: [1 4 7 1 7 1] ⇒ 2

2 2



Amax = 7

Amin = 1

Ans = ~~4~~ 3

Idea :-

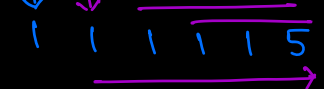
1. For every min ⇒ find the closest (first) MAX on RHS.

2. For every max ⇒ find the closest (first) MIN on RHS.

Code:

- Amax : max(Arr) → O(N)
- Amin : min(Arr) → O(N)

```
ans = INT_MAX / N
for(i=0; i<N; i++) {
    if(A[i] == Amin) {
        for(j=i; j<N; j++) {
            if(A[j] == Amax) {
                ans = min(ans, j-i+1);
                break;
            }
        }
    }
    else if(A[i] == Amin) {
        for(j=i; j<N; j++) {
            if(A[j] == Amin) {
                ans = min(ans, j-i+1);
                break;
            }
        }
    }
}
return ans;
```



TC: $O(N^2)$

SC: $O(1)$

Quiz

A : [8, 8, 8, 8, 8, 8]

Amax = 8

Amin = 8

⇒ (1) [8]

ans = 2

Ex

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

Amin = 1

Amax = 6

minIndex = ~~-1~~ 5, 10

manIndex = ~~-1~~ ~~2~~ 8

ans = ~~0~~ 4 3

Ex

	0	1	2	3	4	5
A :	6	4	1	2	5	6

Amax = 6

Amin = 1

lastMax = ~~-1~~ 0 5

lastMin = ~~-1~~ 2

ans = ~~0~~ 3

Ex

	0	1	2	3	4	5
A :	6	4	1	2	5	1

Amax = 6

Amin = 1

lastMax = ~~-1~~ 0

lastMin = ~~-1~~ 5

ans = ~~0~~ 3

	0	1	2	3	4	5	6	7	8	9	10	11
A:	1	6	4	2	7	7	5	1	3	1	1	5

Amax = 7
Amin = 1

lastMax = 5
lastMin = 10

ans = 3

Code
Todo

TC: O(N)
SC: O(1)

Ex A: [8, 8, 8, 8, 8, 8]

Amax = 8
Amin = 8

lastMax = 5
lastMin = 5

ans = 1
