Carry Forward

Prefix Sum :

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

carry forwarding the prev sum

Q1 Given an array, Build leftMax array

leftmax[i] = max value in the given array from idx 0 to i

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

leftmax[0] = a[0]

leftmax[1] = max(a[0], a[1])

leftmax[2] = max(leftmax[1], a[2])

leftmax[i] = max(leftmax[i-1], a[i])

```
int leftmax[N]
leftmax[0] = a[0]

for(i=1 ; i<N ; i++)
{
    leftmax[i] = max(leftmax[i-1],
                     a[i])
}
```

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

Build rightmax array.

rightmax[i] = Maximum of all values from idx [i, N-1]

	0	1	2	3	4	5	6
A:	-3	6	2	4	5	2	-9
RM	6	6	5	5	5	2	-9

RM[N-1] = A[N-1]

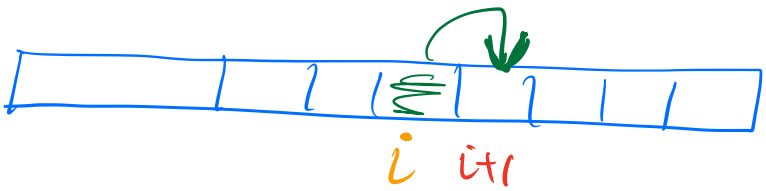
RM[N-2] = max(A[N-1], A[N-2])

RM[N-1]

RM[N-3] = max(RM[N-2], A[N-3])

⋮

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




```
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 Amazon

Given a string of lowercase characters, return the count of pairs (i, j) s.t

$i < j$

$s[i] = 'a'$

$s[j] = 'g'$

} 'ag'

	0	1	2	3	4	5
s:	a	b	e	g	a	g

(0,3) (0,5) (4,5)

ans = 3

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

(0,2) (0,4) (0,6) (5,6)

ans = 4

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

TC: $O(N^2)$ SC: $O(1)$

Obs: Every 'g' will make a valid pair will all the 'a' on the left side

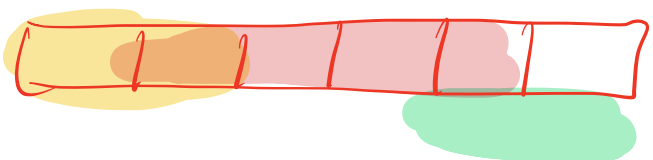
	0	1	2	3	4	5	6	7
Ex	a	c	b	a	g	k	a	g
count_a	1	1	1	2	2	2	3	3
ans	0	0	0	0	2	2	2	5

```
int ans = 0
int count_a = 0

    for (i=0; i<N; i++)
{       if (s[i] == 'a')
            count_a ++
        else if (s[i] == 'g')
            ans += count_a

}

    return ans
```

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

Q3 Given an array, Return the length of smallest subarray which contains both MAX & MIN of the array.

subarray ⇒ continuous part of the array.



A: 1 2 3 4

1, 2

2, 3, 4

1 2 3 4

1, 3, 4 ✗

4

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

max = 6
min = 1

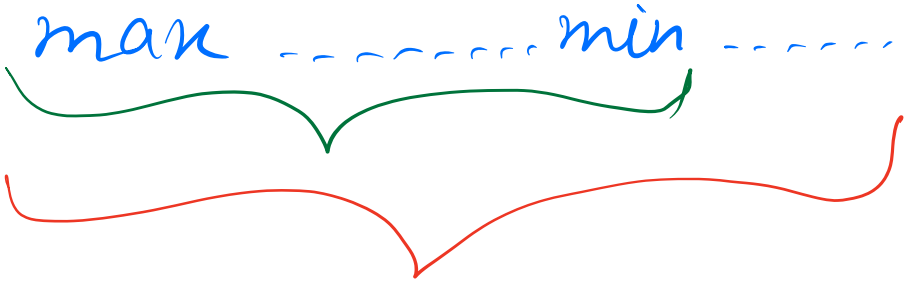
ans = 4

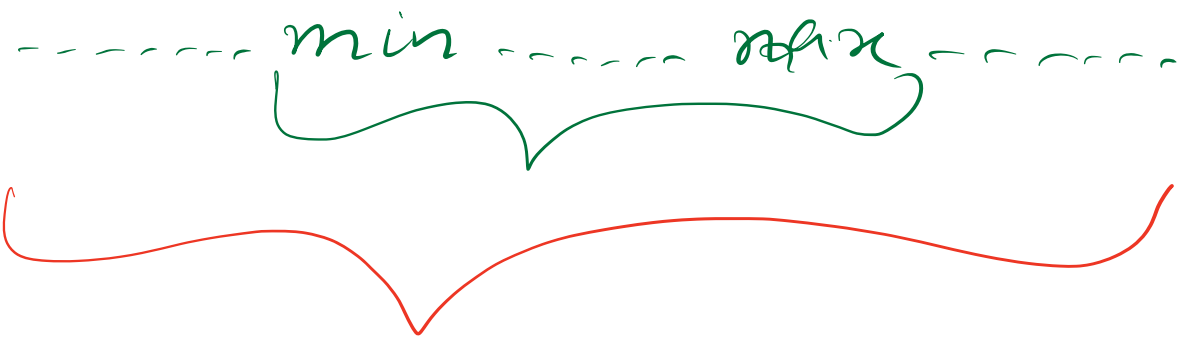
	0	1	2	3	4	5	6	7	8	9	10
A:	2	2	6	4	5	1	5	2	6	4	1

max = 6
min = 1

[8, 10] ⇒ 3

Observation 1 : In the answer subarray, MAX & MIN have to be on the corner.





Obs 2 In the answer subarray there will be only 1 MAX & 1 MIN

Solutions :

min ..... max

max ..... min

→ For every MIN, find the closest MAX on the right side

→ For every MAX, find the closest MIN on the right side

```
ans = INT_MAX   // ∞
// find MAX
// find MIN

for (i=0; i<N; i++)
{   if (A[i] == MAX)
    {   for (j=i; j<N; j++)
        {   if (A[j] == MIN)
            {   ans = min(ans, j-i+1)
                break;
            }
```

j - i + 1 ↓ len of

subarray

```
else if (A[i] == MIN)
{
    for (j = i ; j < N ; j++)
    {
        if (A[j] == MAX)
        {
            ans = min(ans, j - i + 1)
            break;
        }
    }
}
```

j - i + 1 → len of subarray

6 2 3 4 1 6 4 1

ans = 3 for max ... min

ans = 2 for min ... max

TC: $O(N^2)$ SC: $O(1)$

0	1	2	3	4	5
8	8	8	8	8	8

max = 8
min = 8

ans = 1

For each MAX, I want the index of the latest MIN

min ----- max

For each MIN, I want the idx of latest MAX

max ----- min

lat_max → The closest max idx to the left

lat_min → closest min idx to the left

Eg

	-1	0	1	2	3	4	5
		6	4	1	2	5	6
lat_min		$-\infty$	$-\infty$	2	2	2	2
lat_max		0	0	0	0	0	5

↓

INT_MIN

ans = 0 For each iteration

if MAX is found

ans = min (ans, i - lat_min + 1)

min ------ max

idx i

= lat_min

[lat_min, i]

↳ valid

Formula = b - a + 1

= i - lat_min + 1

if MIN is found

ans = min (ans, i - lat_max) + 1

1) Find MAX & MIN

2) lat_max = INT_MIN
   lat_min = INT_MIN
   ans = INT_MAX

3) For each i 0 to n-1

if a(i) == MAX

lat_max = i
ans = min (ans, i - lat_min + 1)

else if (a(i) == MIN)

lat_min = i
ans = min (ans, i - lat_max + 1)

return ans.

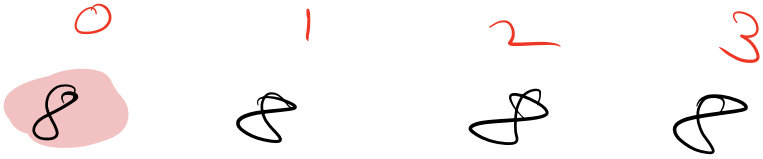
	0	1	2	3	4	5
	6	4	1	2	5	6
lat_min	$-\infty$	$-\infty$	2	2	2	2
lat_max	0	0	0	0	0	5
ans	∞	∞	3	3	3	3

$$ans = \min(\infty, 0 - (-\infty) + 1)$$

$$ans = \min(3, 5 - 2 + 1)$$

$$\min(3, 4)$$

Doubles



last_non 0

last_non 0

ans —

0 - 0 + 1 = 1