SCALER

Array - Carry forward & Subarrays

TABLE OF CONTENTS

Notes

Contest.

↳ 3 Questions

↳ 1.5 hours.

For first 1.5 hours → contest, followed by 1.5 hours contest discussion.

Live Contest → R1 → R2 → R3

19th July. 20th July

Count 'a-g' pairs

< *Question* > : Given a string s of lowercase characters, return the count of pairs (i, j) such that i < j and s[i] is 'a' and s[j] is 'g'.

Eg 1 ⇒ 0 1 2 3 4 5
"a b e g a g"

0,3
0,5
4,5

ans = 3

Quiz → 0 1 2 3 4 5 6
"a c g d g a g"

0,2 5,6
0,4
0,6

ans = 4

Quiz → 0 1 2 3 4 5 6 7
b c a g g a a g

2,3 5,7
2,4 6,7
2,7

ans = 5

BF Idea

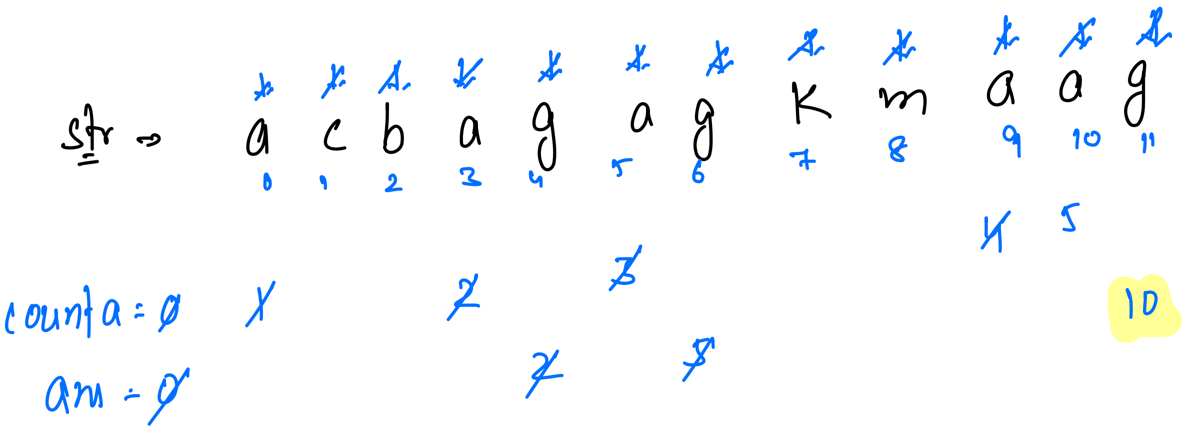
Consider all the pairs and increment the count if you find a valid "a-g" pair.

</> Code

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

T.C → $O(N^2)$
S.C → $O(1)$

Idea Carry Forward the count of a's from l.h.s



</> Code

```
counta = 0 , ans = 0;

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

T.C → O(N)
S.C → O(1)

Subarrays

↳ contiguous part of an array.

Eg → arr[] → [4 1 2 3 -1 6 9 8 12]

0	1	2	3	4	5	6	7	8
4	1	2	3	-1	6	9	8	12

Q → 4 -1 6 9 is sub-array? No.

Q → is 1 a subarray? Yes.

→ Whole array is also a subarray.

→ One single element is also a subarray.

→ Empty array is not a subarray.

Example: arr[ ] → [2 4 1 6 -3 7 8 4]

a. [1, 6, 8]

b. [1, 4]

c. [6, 1, 4, 2]

[✓ d.] [7, 8, 4,]

How to represent a subarray?

→ By specifying start and end index.

→ By specifying start index and length of the subarray.

$$\text{Eg}\rightarrow \text{arr}[\,] \rightarrow \begin{array}{ccccccccc} [4 & 1 & 2 & 3 & -1 & 6 & 9 & 8 & 12] \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array}$$

(st→2, ei→7) ⇒ 2 3 -1 6 9 8

(st→2, len→6) ⇒ 2, 3, -1, 6, 9, 8

Total number of subarrays

arr ⇒ [4, 2, 10, 3, 12, -2, 15]   length = N

0 1 2 3 4 5 6

starting with index 0

ans = 7.

0-0 ⇒ [4]

0-1 ⇒ [4, 2]

0-2 ⇒ [4, 2, 10]

0-3 ⇒ [4, 2, 10, 3]

0-4 ⇒ [4, 2, 10, 3, 12]

0-5 ⇒ [4, 2, 10, 3, 12, -2]

0-6 ⇒ [4, 2, 10, 3, 12, -2, 15]

[4 2 10 3 12 -2 15]   ,N = 7

0 1 2 3 4 5 6

starting with index -1

ans = 6.

1-1 ⇒ [2]

1-2 ⇒ [2, 10]

1-3 ⇒ [2, 10, 3]

1-4 ⇒ [2, 10, 3, 12]

1-5 ⇒ [2, 10, 3, 12, -2]

1-6 ⇒ [2, 10, 3, 12, -2, 15]

Total number of subarrays

$$\text{arr} \Rightarrow \begin{array}{ccccccc} [4, & 2, & 10, & 3, & 12, & -2, & 15] \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 \end{array} \quad \underline{\underline{N}}$$

starting from index 0 ⇒ N +

starting from index 1 ⇒ N - 1 +

starting from index 2 ⇒ N - 2 +

starting from index 3 ⇒ N - 3 +

———— → 4 ⇒ N - 4 +

———— → 5 ⇒ N - 5 +

|

|

———— N - 2 ⇒ 2 +

———— N - 1 ⇒ 1

total no. of subarrays → $\frac{N(N+1)}{2}$

< Question > : Given an array, si and ei. Print from si to ei.

si ≤ ei

arr → [4 2 10 3 12 -2 15]   si = 2, ei = 5

0 1 2 3 4 5 6

o/p. → 10 3 12 -2

```
• void printSubarray( arr, si, ei ) {
      for( K = si;  K ≤ ei ; K++){
          print(arr[K]);
      }
  }
```

→ [T.C → O(N)
S.C → O(1)]

print 1 subarray → T.C 0(N)

$$\text{print } \frac{N(N+1)}{2} \text{ subarrays} \Rightarrow \frac{N(N+1)}{2} \times N \Rightarrow \frac{N^3}{2} + \frac{N^2}{2}$$

$$\Rightarrow \underline{\underline{O(N^3)}}$$

< Question > : Print all the possible sub-arrays of the given array.

$1 \le N \le 10^2.$

[5, 7, 3, 2]

0 1 2 3

0-0	O/P - [5]
0-1	[5, 7]
0-2	[5, 7, 3]
0-3	[5, 7, 3, 2]
1-1	[7]
1-2	[7, 3]
1-3	[7, 3, 2]
2-2	[3]
2-3	[3, 2]
3-3	[2]

Idea Consider all the subarrays & print Subarray( )

</> Code

```
for( si = 0; si < N; si++){
    for(ei = si; ei < N; ei++){
        printSubArray( arr, si, ei);
    }
}
```

T.C → $O(N^3)$

S.C → $O(1)$

arr[] → [10 3 9 12 8]

10	3	9	12	8
0	1	2	3	4

0, 0
0, 1
0, 2
0, 3
0, 4

1, 1
1, 2
1, 3
1, 4

2, 2
2, 3
2, 4

Min Max

< *Question* > : Given an array of N integers, return the length of smallest subarray which contains both maximum and minimum elements of the array.

1 ≤ N ≤ 10^6

arr[ ] → [2 2 6 4 5 1 5 2 6 4 1]

0 1 2 3 4 5 6 7 8 9 10

max → 6
min → 1

ans=3

arr[ ] → [1 2 3 1 3 4 6 4 6 3]

0 1 2 3 4 5 6 7 8 9

ans=4

arr[ ] → [8 8 8 8 8 8]

0 1 2 3 4 5

max → 8
min → 8

ans=1

Idea – Consider all the subarrays. If a subarray contains min & max element, then length of that subarray can be a potential ans.

T.C → $O(N^3)$
S.C → $O(1)$

Observation

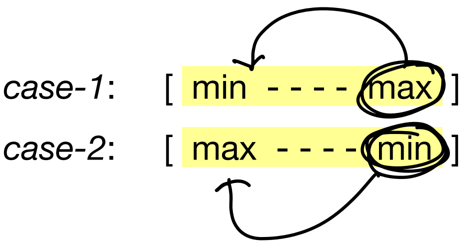
1. There must be exactly one occurrence of min & max element.

[min min - - - - max]

2. Min and max elements should be the end point of subarray.

Closesct min ele. on l.h.s.

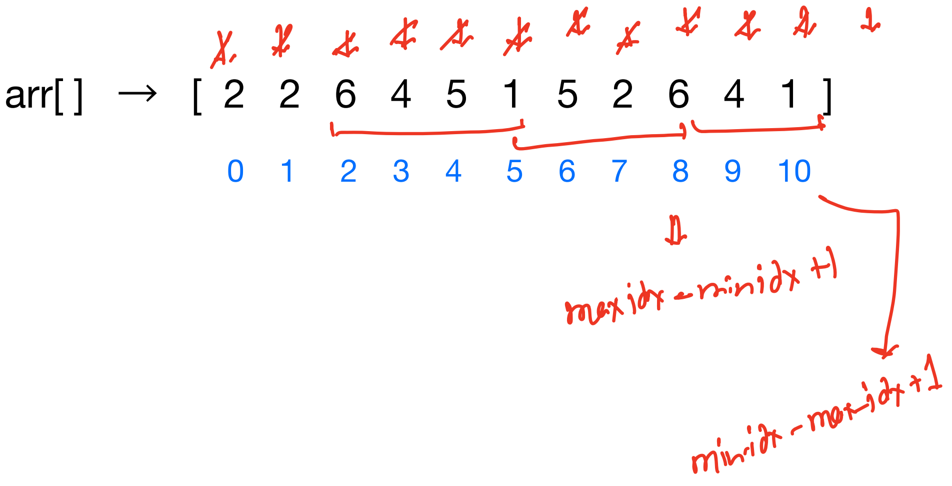
3.



case-1: [min - - - - max]

case-2: [max - - - - min]

Closest max ele. on l.h.s.



arr[ ] → [2 2 6 4 5 1 5 2 6 4 1]

0 1 2 3 4 5 6 7 8 9 10

min → 1

max → 6

min - idx = ~~-1~~ ~~5~~ 10

max. idx = ~~-1~~ ~~2~~ 8

ans = ~~11~~ ~~4~~ 3

</> Code

```
//1. Find min & max element

min = arr[0],  max = arr[0];

for( i = 1; i < N; i++){
        min = Min( min, arr[i]);
        max = Max (max, arr[i]);
}

//2. Carry forward min-idx , max-idx.

ans = N,  min-idx = -1,  max-idx = -1
for( i = 0; i < N; i++){
        if( arr[i] == min){
                min-idx = i;
                if( max-idx != -1){
                        ans = Min( ans,  i - max-idx +1);
                }
        }
        else if ( arr[i] == max){
                max-idx = i;
                if (min-idx != -1){
                        ans = Min(ans,  i - min-idx +1);
                }
        }
}
return ans;
```

T.C → O(N)
S.C → O(1)

→ S.W

→ Contribution

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

$$\underbrace{psum[N-1] - psum[0] \;==\; 0}$$

⇒ Revision