Arrays:
Carry Forward

↳ carrying information forward

Vinay Neekhra

Senior Instructor & Mentor

Reachable in Scaler Lounge

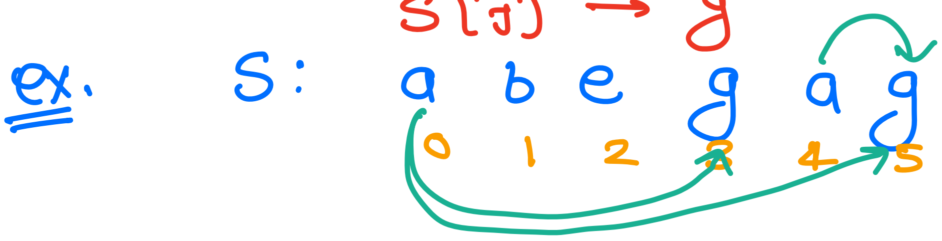
"First show up, then do it, then do it right,
then only do it better"

Q. Given a string of lowercase alphabets. Return count of (i,j), such that AMAZON

$i < j$

$S[i] \to$ 'a'

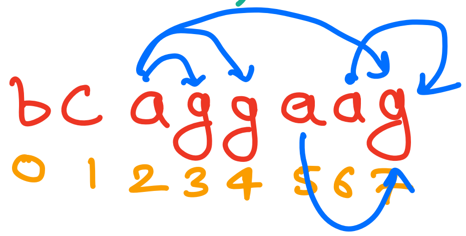
$S[j] \to$ 'g'



ex. S: a b e g a g ans = 3

0 1 2 3 4 5

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



Quiz : bcaggaag ans = 5.

0 1 2 3 4 5 6 7

10year old kid soln.

$N \in [1, 10^5]$

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

```
|   |   |       if ( s[j] == 'g'){
|   |   |       |       ans += 1
|   |   |       }
|   |   }
}   }
return ans;
```

T.C = O(N²) — TLE

S.C = O(1)

Obsⁿ. (I) starting index always contains 'a'

N	String	ans
0	[ ]	0
1	[b]	0
1	[x]	0
2	[a b]	0
2	[g a]	0
2	[a g]	1
3	[a b g]	1
3	[a a g]	2
3	[a g b]	1
3	[a g a]	1

3 [agg] → 2

4 [aaag] 3

4 [aagg] → 4

[abagcdagg] → 8

ex.	a	c	b	a	g	k	a	g	g	+3
	0	1	2	3	4	5	6	7	8	
count 'a'	1	1	1	2	2	2	3	3	3	
ans	0	0	0	0	2	2	2^{+3}	5^{+3}	8	

a	b	a	g	a
0	1	2	3	4

ans ~~0~~ 2

count a ~~0~~ ~~1~~ ~~2~~ ~~2~~ 3

i ~~0~~ ~~1~~ ~~2~~ ~~3~~ ~~4~~ 5

```
ans = 0
count_a = 0

for (int i=0; i<N; i++) {
        if (s[i]=='a') {
            count_a += 1
        }
```

```
    else if (s[i] == 'g') {
        ans += count_a
    }
}
return ans;
```

T.C. = O(N)
S.C. = O(1)

H.W try to do this using count_g

Q. Given an array. Return the length of smallest subarray which contains both the max & min of the array.

Subarray : any contiguous chunk of the array

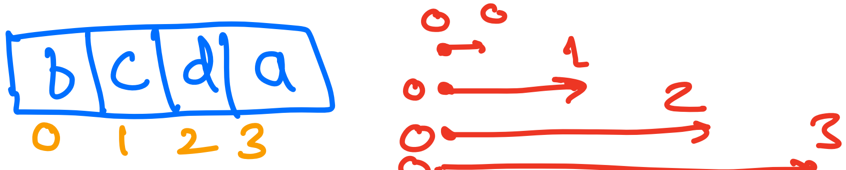
2 | 1 | 3 | 4 ⇒ [2, 4] ×

⇒ [1,3,4] ✓

⇒ [2] ✓

[L, R]
⇒ R-L+1

③

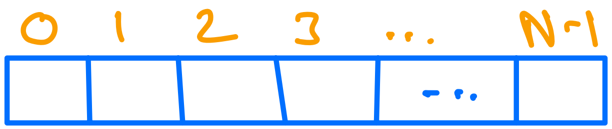


Sub arrays which starts with index '0' ⇒ 4

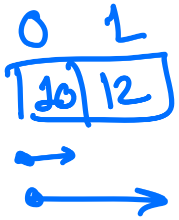
[[b], [bc], [bcd], ([bcda])] → subarray

Sub arrays which starts with index '1'

⇒ [[c], [c,d], [c,d,a]] ⇒ 3



⇒ N size



Subarrays starting with index '0' ⇒ N

+

Subarrays starting with index '1' ⇒ N-1

+

Subarrays starting with index '2' ⇒ N-2

+

⋮

+

Subarrays starting with index 'N-1' ⇒ 1

count of all subarrays ⇒ $\frac{N*(N+1)}{2}$

$= O(N^2)$

ex 1

[1, 2, 3, 1, 3, 4, 6, 4, 6, 3]

0 1 2 3 4 5 6 7 8 9

Array max → 6

Array min → 1

ans = 4

Quiz

ans = 3

[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

Brute force: for every subarray, check if it contains both max & min

① find max & min O(N) × → O(N²)

② for every subarray, check if it contains max & min (O(N))

T.C = O(N³)

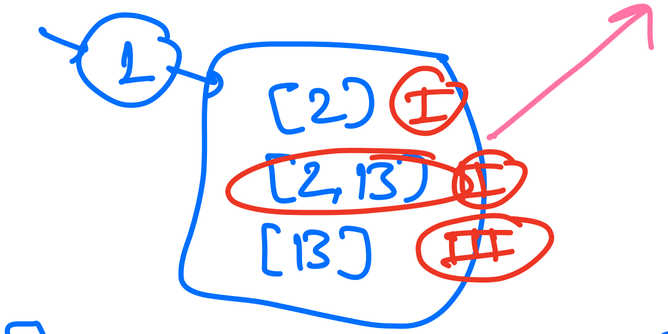
S.C = O(1)

SFF expected

output

Obsn

N	Array	ans
0	[ ]	-1 / 0
1	[4]	1
2	[2, 13]	2
2	[13, 15]	2
2	[10, 10]	1
3	[10 15 19]	3
3	[19, 10, 15]	2
3	[10, 10, 15]	2



3 [11, 10, 15] → 2

4 [50, 9, 7, 18] → 3

4 [5, 5, 7, 7, 19, 11, 18, 19] → 4

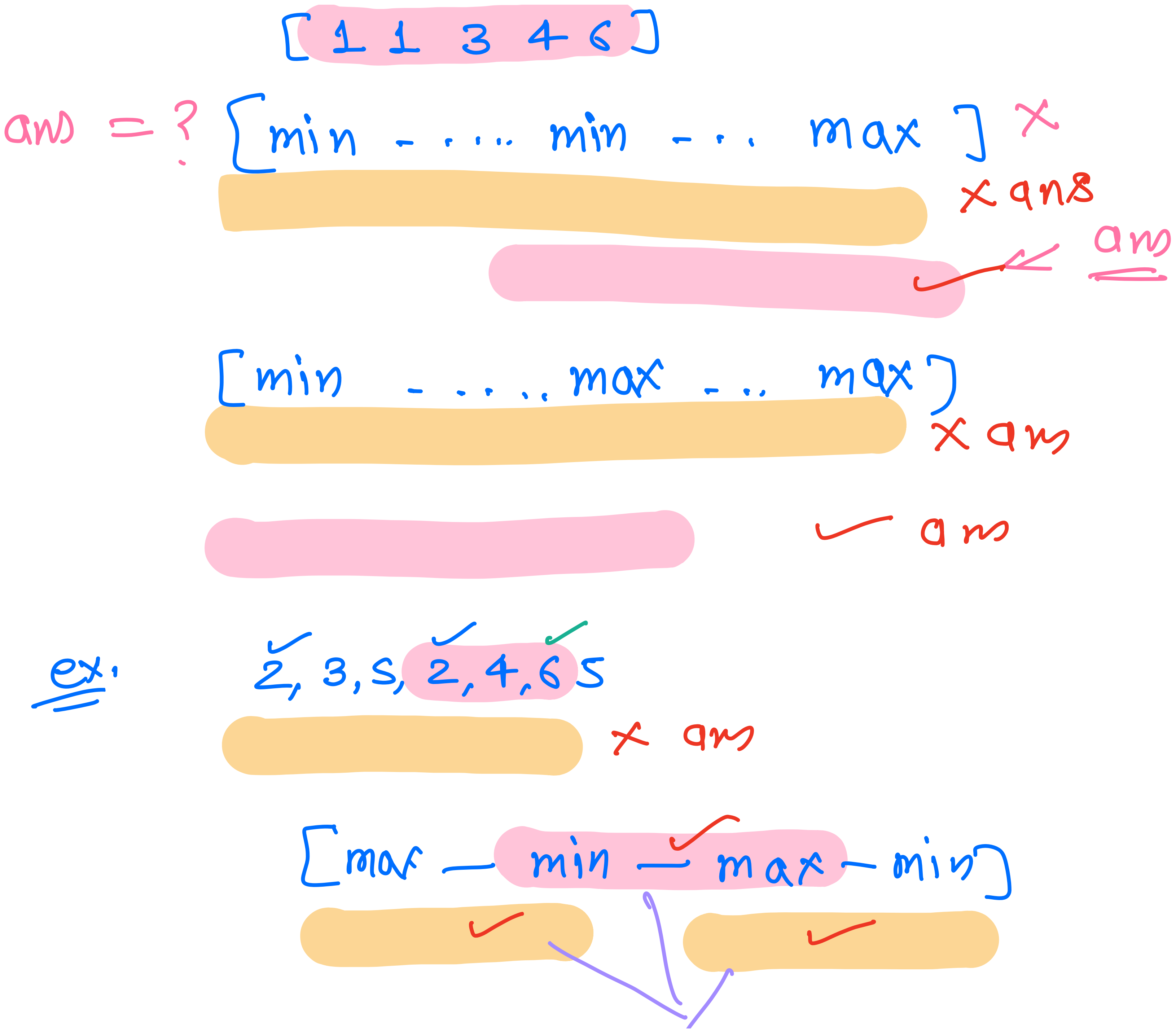
Obs^{n} (I) subarray corners should be (max, min)

[mid, max ____ min] (max, min)

[mid, min -.. max] (min, max)

✗

(II) there would be exactly one min and one max in our answer (smallest length subarray)



[1 1 3 4 6]

ans = ? [min ..... min ... max] ✗

✗ ans

✓ ans

[min ...... max ... max]

✗ ans

✓ ans

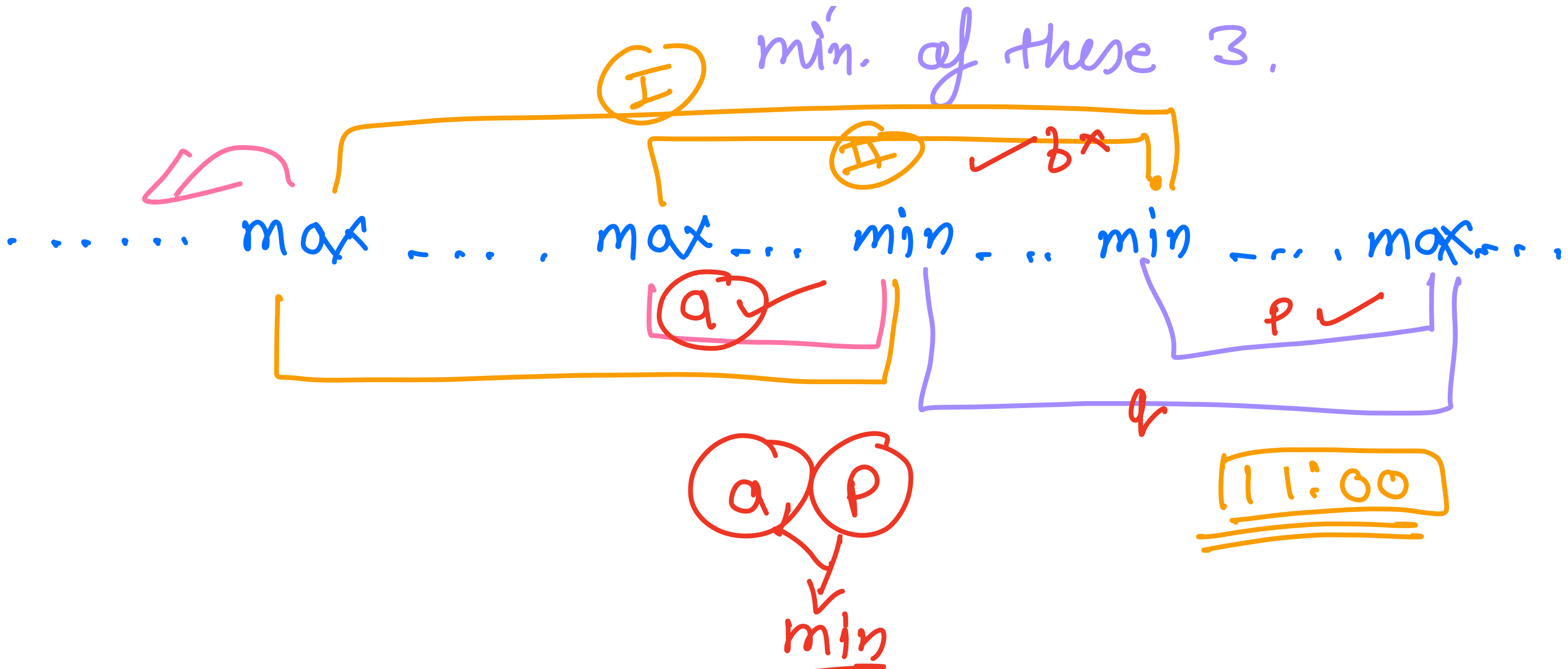
ex:

✓ 2, 3, 5, ✓ 2, 4, ✓ 6 5

✗ ans

[max — min — ✓ max — min]

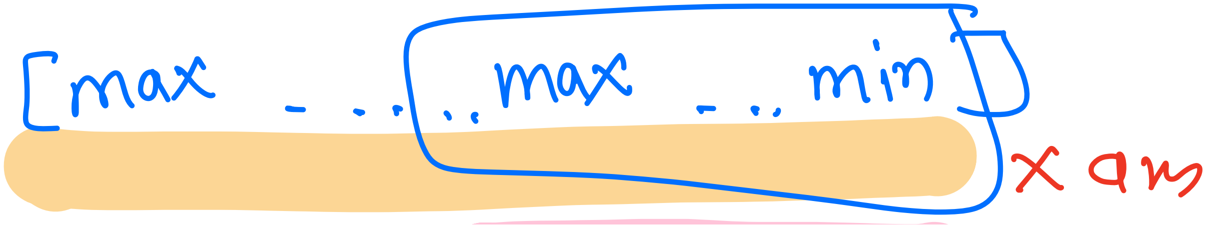
✓ ✓


min. of these 3.
I
II
3x
. . . . . . max . . . . max . . . min . . . min . . . max . . .
a
p
q
a
p
min
11:00

going left to right

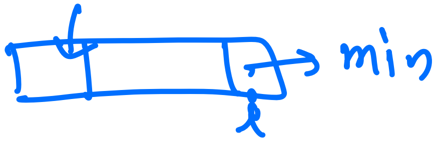
→ max ⇒ latest min before

→ min ⇒ latest max before



✓ ans [5 4 3 -1 2 0]

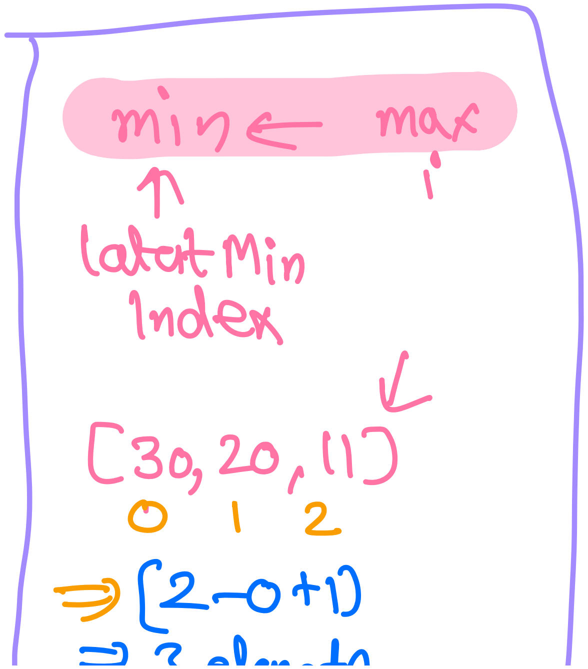
```
// find the max  - Amax
// find the min  → Amin

→ latestMinIndex = ~~0~~ -1   -1 denotes ( Haven't found any max)
  latestMaxIndex = ~~0~~ -1

  ans = N

 for (int i=0; i<N; i++) {
        If (A[i] == Amin) {
            latestMinIndex = i
            If (latestMaxIndex >= 0){
                ans = min( ans,
                           (i - latestMaxIndex + 1) )
```



```
                }
      ~~else~~ if ( A[i] == Amax){
                latestMaxIndex = i
                if (latestMinIndex >= 0){
                    ans = min(ans,
                              (i - latestMinIndex+1))
                }
          }
    }
return ans;
```

$TC = O(n)$

$SC = \underline{\underline{O(1)}}$



→ 3 element

✗ ✗ ✓ ✗ ✗ ✓ ✗ ✗ ✓ ✗ ✓

A = 2, 2, 6, 4, 5, 1, 5, 2, 6, 4, 1 , N = 11

0 1 2 3 4 5 6 7 8 9 10

Amax = 6, Amin = 1

Lmin = ~~-1~~ ~~5~~ 10

Lmax = ~~-1~~ ~~2~~ 8

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

i = ~~0~~ ~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~ ~~7~~ ~~8~~ ~~9~~ 10

ans = 3

✓

A = 7, 7, 7, 7, 7

0 1 2 3 4

Lmin = ~~-1~~ ~~0~~ ~~1~~ ~~2~~ ~~3~~ 4

Lmax = -1

ans = 5

i = ~~0~~ ~~1~~ ~~2~~ ~~3~~ ~~4~~ 5

ans = 1

Amin = 7, Amax = 7

⇒ if (Amax == Amin)

return 1;

Q. Give an array, find no. of leaders.

leader: an element which is greater than all the elements on the left side.

A[i] > [0, i-1]

ex.

✓ ✓

2, 5, 3, 4, 17, 16

0 1 2 3 4 5

[2,5,17] , ans = 3

H.W. ⇒ Using carry forward to find the optimal sol^n

———x———

Doubt Session

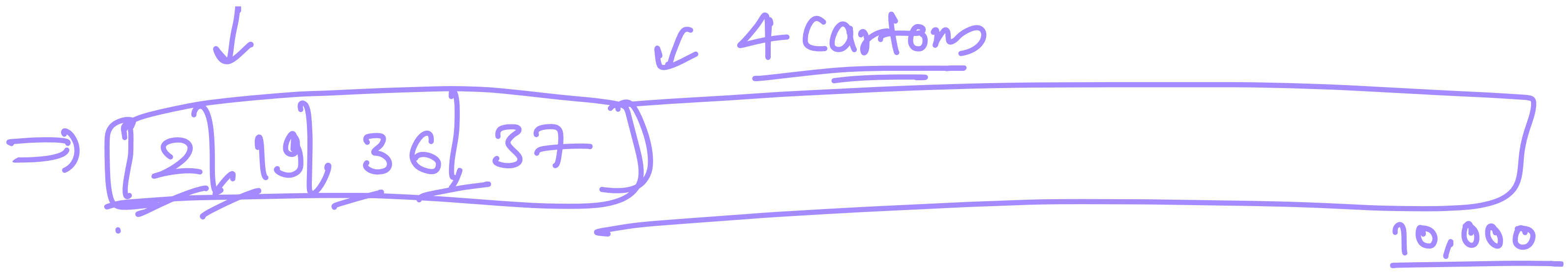
Q1 Yes, it is acceptable

C++, 1 sec

pyth 3 sec

✓ ✓ ✗

Q2 [2, 17, 17, 1] —————]10,000



Space req^n $\propto$ input (N)

O(N)