14/01/22

DSML Intermediate – DSA

Arrays – 3

Today's Content : Carry Forward Technique

Q1 Count pairs "AG"

Q2 Count leaders

Q3 Smallest Continuous Part Contains Min & Max

Concept → Why TLE?? Importance of Constraints.

Q5* Bulbs

- Recap
- HW
- * Special Index

M 17 – SqLarray

W 19 – 2D Arrays ✓

F 21 – Interview Problems.

* Pragy Agarwal

Q1 Count Pairs 'AG'

collection of chars = array of characters.

Given a string s, calculate no. of pairs (i,j) such that

$i < j$ and $s[i] = 'a'$ and $s[j] = 'g'$.

Eg 1.

	0	1	2	3	4	
"	b	a	a	g	c	"

=> (1,3)
(2,3)

ans = 2

NOTE: All characters are lower case 'a'-'z'.

Eg 2

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

a) 5
b) 4
c) 2
d) 6

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

Brute Force

```
cnt = 0
∀i
∀j
    if s[i] = 'a' and s[j] = 'g'
        cnt += 1.
```

```
cnt = 0
n = len(s)
for i in range(0, n):
    for j in range(i+1, n):
        if s[i] == 'a' and s[j] == 'g':
            cnt += 1.
```

→ n

→ $n-(i+1) = n-i-1$

TC: $O(N^2)$

SC: $O(1)$

i	j	cnt
0	[1,n]	n-1
1	[2,n]	n-2
⋮	⋮	⋮
N-1	[N-1,N]	1
		N(N-1)/2

Q1 Optimize

```
cnt=0
n= len(s)
for i in range(0,n):

    if s[i] == 'a' :
        for j in range(i+1, n):
            if s[j]=='g' :
                cnt += 1.
```

cnt of g's from (i+1) to n.

N = 8

0 1 2 3 4 5 6 7

a a a a a a a g

N-1

N-2

1

⇒ $O(N^2)$

Eg.

cnt = 0

a d g a g a g f g

✓ ✗ ✗ ✓ ✗ ✓ ✗ ✗ ✗

cnt += 4

cnt += 3

cnt += 2

Cnt = 9

Cnt g's from end to this index.

* Do not use extra space!

Q1

0	1	2	3	4	5	6	7	8
a	d	g	a	g	a	g	f	g
ans+=c	×	c+=1	ans +=c	c+=1	ans+=c	c+=1	×	cnt_g+=1
ans=9		c=4	ans =5	c=3	ans=2	c=2		c+=1
								c=1

cnt_g = 0 (c for short)
c

ans = 0

Carry Forward Technique

Pseudo Code

```
N = len(s)
ans = 0
c = 0  # cnt of g's to the right
for i in range(N-1, -1, -1):
    if s[i] == 'g':
        c += 1
    elseif s[i] == 'a':
        ans += c
```

for(i=N-1; i>=0; i--)

Time = O(N)

Space = O(1)

range(0,5) = [0, 1, 2, 3, 4]

range(6, 0, -1) = [6, 5, 4, 3, 2, 1] ⇒ 0 is not included

range(6, -1, -1) = [6, 5, 4, 3, 2, 1, 0]

excluded

Todo

HW

Carry Forward no. of a's

if s[i] == 'g', no. of a's on the left side.

Q2 Leaders in an Array

Given an array A of size N, you have to count the no. of leaders in this array A.

An elt is a leader if it is strictly greater than all the elements to its right side.

* NOTE: A[N-1] is always considered a leader.

Eg.

0	1	2	3	4	5	6	7
15,	-1,	7,	2,	5,	4,	2,	3
✓	✗	✓	✗	✓	✓	✗	✓

Leaders: 5

Eg

0	1	2	3	4	5	6
16,	17,	17,	4,	3,	5,	2
✗	✗	✓	✗	✗	✓	✓

cnt = 3

Brute $\forall i$ check if A[i] is a leader

A[i] > max from A(i+1, n-1)

$i = n-2 \Rightarrow i+1 = n-1$

$i = n-1 \Rightarrow i+1 = n$ → Last.

```
→ cnt = 1 # Last elt is leader
for i in range(0, n-1):
    maxi = A[i+1]
    for j in range(i+1, n):
        maxi = max(maxi, A[j])
    if A[i] > maxi:
        cnt += 1
```

max from (i+1) to n-1 right of i

Time = $O(N^2)$

Q2

cnt = 1

6, 5 maxi

	0	1	2	3	4	5	6
Ex	8	-2	4	7	6	5	1
		maxi = ~~-2~~	4	7	7	7	7

get max

i = 0

8 > 7

⇒ cnt = 2

CF Approach (CFA)

0	1	2	3	4	5	6
8	-2	4	7	6	5	1
C = 5, maxi = 8	maxi = 7	maxi = 7	C = 4, maxi = 7	C = 3, maxi = 6	C = 2, maxi = 5	✓ C = 1

maxi = 1

```
N = len(A)
cnt = 1   # c
maxi = A[N-1]   # store the max of all elts to right.
for i in range(N-2, -1, -1):
    if A[i] > maxi:
        cnt += 1   # A[i] is a leader.
        maxi = A[i]
return c
```

Time = O(N)

space = O(1).

Q2 $A[3] = [10, 8, 8]$ (i)

$c = 1$, $maxi = 8$ Initial.

$c = 2$, $maxi = 10$

---

* Subarray : Continuous part of an Array

NOTE: Single element ✓ ✓

: Entire ✓ ✓

: Empty array ✗

Splice

a[3:7]

From index 3 to 6

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

✓ [8, -3, 0, 9, 10] => [2, 6] index 2 to 6 is a continuous part ✓

✗ [3, 2, 0, 9, 8] => [0, 1], [4, 5], [7, 7]

NOT continuous. 3 diff. part.

Quiz $A = [4, 5, 1, 9, 0, 2, 3, 5]$ (i — j)

[i, j] All elements from index i to j are included.

a) [5] ✓

b) [4, 5, 1, 0] ✗

c) [9, 0, 2, 3] ✓

d) [4, 5, 1] ✓

privilege

sort( ) → To sort N numbers Time = O(N log N)

max(a, b) → return max of 2 numbers = a, b Time = O(1)

min(a, b) → return min of 2 numbers = a, b Time = O(1)

Time taken by inbuilt methods should be known.

---

min(A)
max(A) Time = O(N)

A is an array

len(A) = O(N)

sum(A) = O(N)

Q3 Closest Min Max

- Direct I
- Google
- Goldman

Given an array A, find the length of the smallest subarray which contains both min & max of the array.

Eg.

0	1	2	3	4	5	6	7	8	9
1,	2,	3,	1,	3,	4,	6,	4,	6,	3

$A_{min} = 1$

$A_{max} = 6$

$A_{max} = max(A)$

$A_{min} = min(A)$ } $O(N)$ time.

ans = 4

[3, 6] = A[3:7]

Code

7 is excluded.

on paper

$6-3+1 = 4$

Eg 2.

0	1	2	3	4	5	6	7	8	9	10
2,	2,	6,	4,	5,	1,	5,	2,	6,	4,	1

len = 3

$A_{max} = 6$

$A_{min} = 1$

$[8, 10] = 10 - 8 + 1 = 3.$

Eg. 3

0	1	2	3	4	5	6	7	8	9	10
1,	6,	4,	2,	7,	7,	5,	3,	1,	1,	5

$A_{max} = 7$

$A_{min} = 1$

$[5, 8] = 8 - 5 + 1 = 4.$

- 3
- 4
- 5
- 6

Q3

6 mins break

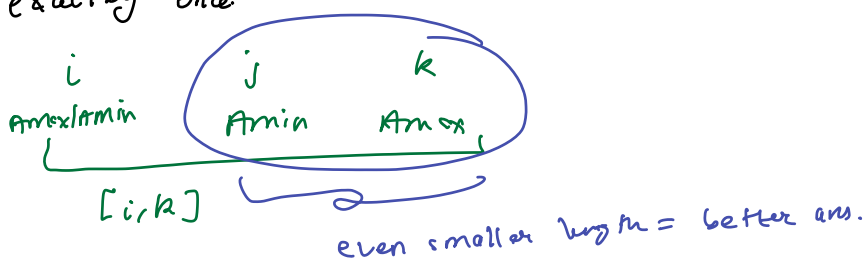
$\boxed{10.35}$

→ Try to brute force

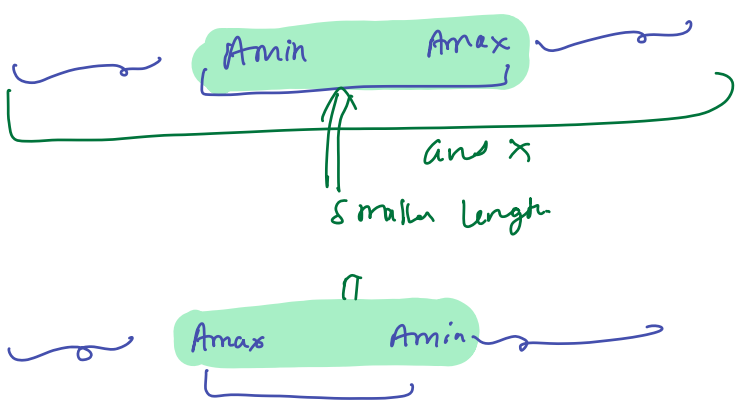
→ Figure observations

Observations

① In your ans subarray, it should only contain Amin & Amax exactly once



② Amin & Amax would be present at the ends of the answer subarray.



Q3

i) $A_{min} = 1$

$A_{max} = 6$

$A_{min} = \min(A)$

$A_{max} = \max(A)$

ii) $ans = N = 14$

0	1	2	3	4	5	6	7	8	9	10	11	12	13
2	2	6	4	5	1	5	2	6	4	3	4	1	3
X	X		X	X		X	X		X	X	X		X

A_{max}

Find index of 1st mini on its right side

$\min i$

[2, 5]

$len = 5 - 2 + 1$

$= 4.$

$ans = \min(ans, 4)$

$= 4.$

A_{min}

Find idx of 1st maxi on its right side. $\rightarrow \max i$

$\max i = 8$

[5, 8]

$8 - 5 + 1$

$= 4$

A_{max}

$\min i$

$= 12$

[8, 12]

$12 - 8 + 1$

$= 5$

A_{min}

$\max i$ X

$ans = 4$

0	1	2	3	4	5	6	7	8	9	10	11	12	13
2	2	6	4	5	1	5	2	6	4	3	4	1	3
X	X		X	X		X	X		X	X	X		X

[5, 8]

$8 - 5 + 1$

$= 4$

Pseudo Code

```
N = len(A)
ans = N
Amin = min(A)
Amax = max(A)
if (Amin == Amax) return 1;
for i in range(0, N):
    if A[i] == Amax:
        for j in range(i+1, N):
            if A[j] == Amin:
                ans = min(ans, j-i+1)
                break
    else if A[i] == Amin:
        for j in range(i+1, N):
            if A[j] = = Amax:
                ans = min(ans, j-i+1)
                break
return ans
```

Edge case.

0 1

[8, 8, 8, 8, 8]

- Amin = 8
- Amax = 8.

ans = 1

~~ans~~ = 2

iterate and set mini (1st index on right) A[mini] = Amin

iterate and get maxi (1st index on right) A[maxi] = Amax

Hint: Try to go from right to left & carry forward the mini & maxi (min index on right, max index on right)

Amax = 6
Amin = 1

ans = N = 13
→ mini = None
→ maxi = None → ?

0	1	2	3	4	5	6	7	8	9	10	11	12
1	6	4	6	5	1	5	2	6	4	4	2	1
↓	↓	×	↓	×	↓	×	×	↓	×	×	×	mini = 12
mini = 0	maxi = 1		maxi = 3		mini = 5			maxi = 8				maxi = None
maxi = 1	mini = 5		mini = 5		maxi = 8			mini = 12				
[0, 1]	[1, 5]		un = [3, 5]		[5, 8]			[8, 12]				
len = 2	un = 5		5-3+1 = 3		un = 8-5+1 = 4			un = 12-8+1 = 5				
ans = 2	ans = 3		ans = 3		ans = 4			ans = min(ans, len) = min(13, 5) = 5				

Ce3

0	1	2	3
8,	8,	8,	8

min=2

maxi= None

mini=3

maxi = None

ans = 4

mini = None

maxi = None

```
N = len(A)
ans = N
Amin = min(A)        → O(N)
Amax = max(A)        → O(N)
mini = None, maxi = None.
```

$\Rightarrow 3N$

$O(N)$.

Edge case.

```
if (Amin == Amax) return 1;
for i in range(N-1, -1, -1):        → O(N)
    if A[i] == Amax: # 1st min on right side
        maxi = i
        if mini != None: # There is a mini on right
            # [maxi, mini]
            ans = min(ans, mini-maxi+1)   # update for better ans
    else if A[i] == Amin: # 1st max on right side.
        mini = i
        if maxi != None: # There is a maxi on right
            # [mini, maxi]
            ans = min(ans, maxi-mini+1)   # update
return ans
```

Time = O(N)

Space = O(1)

TODO: Try to solve it using a forward iteration

1st min/max to the left side.

```
def min_custom(a, b):
    if a > b:
        return b
    else
        return a
```

$\Rightarrow TC = O(1)$

Q5*

Why do get TLE?

TLE??

5 min

Hiring Challenges → understand → observation → Code

Google

1 hr Coding Test. → Q1, Q2, Q3

1.30 hr

TLE ← TLE (Optimization 1)

TLE (Optimization 2)

Online Editors → Execution Time 1 sec

Judge

Python 5 sec.

Run on their servers

1 GHz

1 GHz = 10^9 instructions/sec.

Observation: Our code can have atmost 10^9 instructions.

```
for i in range(0, 100):
    print(i)
```

```
for (i=0; i<100; i++){
    print(i)
}
```

1 1 $i=i+1$ 2 1

100 iterations ⇒ approx 400 instructions

4 instructions per loop

In our code,

1 iteration = 10 instructions.

1 iteration = 100 instructions.

max iterations possible

→ $\frac{10^9}{10}$ instructions = 10^8 iterations

→ $\frac{10^9}{100} = 10^7$

Key lie ⟹ $[10^7 - 10^8]$ iterations

Threshold

Constraints

$1 \le N \le 10^5$ → array size

$1 \le arr[i] <= 10^9$

Question: No of leaders.

Sol1 $O(N^2)$

$N = 10^5 \approx N^2 = 10^{10}$ iterations.

TLE

```
i=0 → n :
    j=i+1 → n :
        -
```

Sol2 $O(N)$

$N = 10^5 \to N = 10^5$ iterations.

Correct Answer

Constraints

$1 \le N \le 10^3$

Approach: $O(N^2)$ Brute Force.

No. of iterations $= N^2 = 10^6$

Doubts

Iterate

mini = −1

mini = max(A) → +∞

maxi = min(A) → −∞

Python

minOdd , maxEven

−97 = 54

−97 % 2 = 1

4 % 2 = 0

−98, 54, −52, 15, 23, −97, 12, −64, 02, 80