TCS NQT

Programming Problems with implementations

Stacks & Queues-02

Lecture 10

By- Aditya sir

About Aditya Jain sir

1. Appeared for GATE during BTech and secured AIR 60 in GATE in very first attempt - City topper
2. Represented college as the first Google DSC Ambassador.
3. The only student from the batch to secure an internship at Amazon. (9+ CGPA)
4. Had offer from IIT Bombay and IISc Bangalore to join the Masters program
5. Joined IIT Bombay for my 2 year Masters program, specialization in Data Science
6. Published multiple research papers in well known conferences along with the team
7. Received the prestigious excellence in Research award from IIT Bombay for my Masters thesis
8. Completed my Masters with an overall GPA of 9.36/10
9. Joined Dream11 as a Data Scientist
10. Have mentored working professions in field of Data Science and Analytics
11. Have been mentoring GATE aspirants to secure a great rank in limited time
12. Have got around 27.5K followers on Linkedin where I share my insights and guide students and professionals.

Telegram

Aditya Jain PW

Telegram Link for Aditya Jain sir:

https://t.me/AdityaSir_PW

(Q.1) Next Greater element on right.

-1

a = [4, 2, 5, 17, 12, 2]

ans = [5, 5, 17, -1, -1, -1]

eg2 :- a = [1, 4, 2, 17, 9, 12]

res = [4, 17, 17, -1, 12, -1]

eg3 :- a = [1, 2, 4, 9, 12, 17]

res = [2, 4, 9, 12, 17, -1]

eg4 :- a = [17, 12, 9, 4, 2, 1] desc

res = [-1, -1, -1, -1, -1, -1]

Soln -1 :

res = [-1, -1, . . . -1]

```
for i = 0 ⟶ n-1
{
    for j = i+1 ⟶ n-1
    {
        if (a[j] < a[i])
        {
            res[i] = a[j]
            break;
        }
    }
}
```

TC

$O(n^2)$

Better Soln :-

Stack → Recent, Next, Nearest

push()
pop()
top()

a = [1, 4, 2, 17, 9, 12]

Loader

1) L ← R

2) while (top stack ≤ curr-elem)

pop()

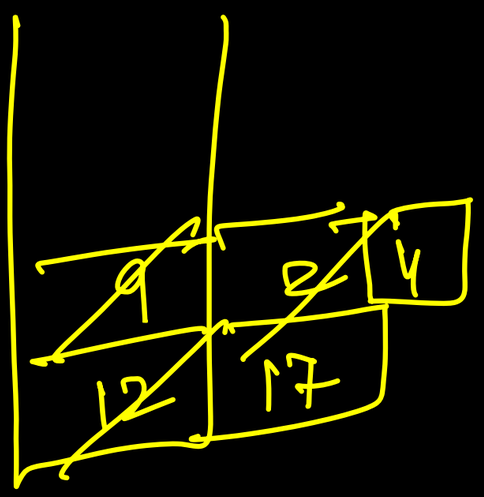
3) if (stack empty)

push(-1).

4) push(a(i))

Dry Run :-

a = [1, 4, 2, 17, 9, 12]



12 → -1

9 → top() → 12

17 → 9 < 17 → pop

12 < 17 → pop.

2 ⟶ top 1772 ⟶ 17

4 ⟶ 2 < 4 ⟶ 17 > 4 ⟶ 17

pop

1 ⟶ top : 4 > 1 ⟶ 4

q : [1, 4, 2, 17, 9, 12]

res : [4, 17, 17, -1, 12, -1]

```
int res[n]

    Stack<int> s;

    for (i = n-1 ------> 0)
        {
            while (stack not empty and
                   top <= a(i))
            {
                pop()
            }
        }
    if ( stack is empty)
        {
            res(i) = -1
        }
```

else {
res(i) = top
}

O(n)

push(a[i])

}

(Q.2) Maximum Consecutive ones after flipping atmax k zeros.

→ binary arr ⟶ a[] a(i) = 1/0

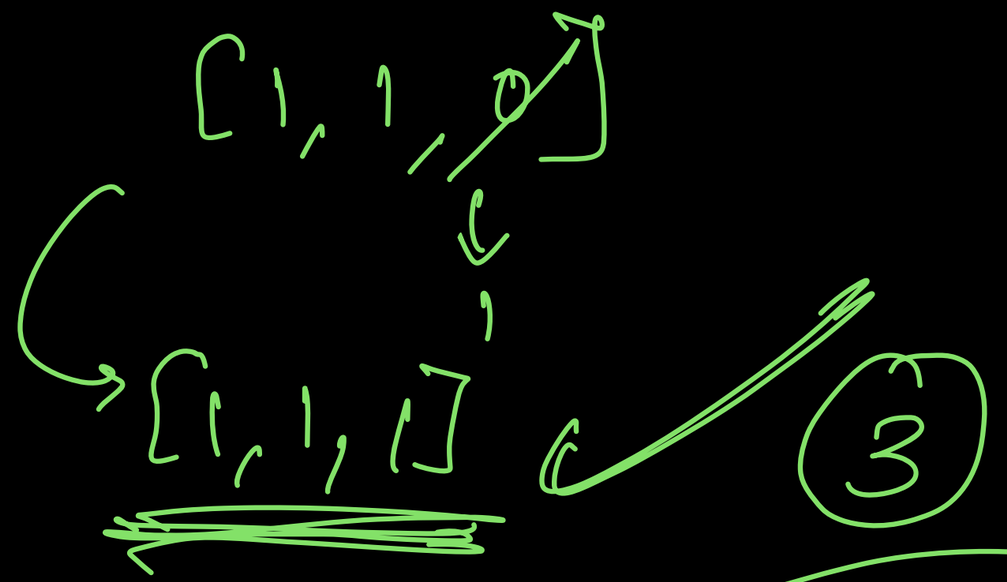
int k

Find max length of subarray with all 1's after flipping at max k 0's to 1's. 0 ⟶ 1

eg1 :- $a = [1, 1, 0]$

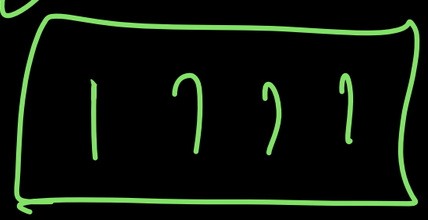
$k = 1$

ans = 3



eg2 :- $a = [1, 0, 0, 1, 0, 1, 0, 1, 0]$ $k = 2$

ans : 5



5

Logic : Appr1 : Brute force .

Logic :- 1) find all subarrays

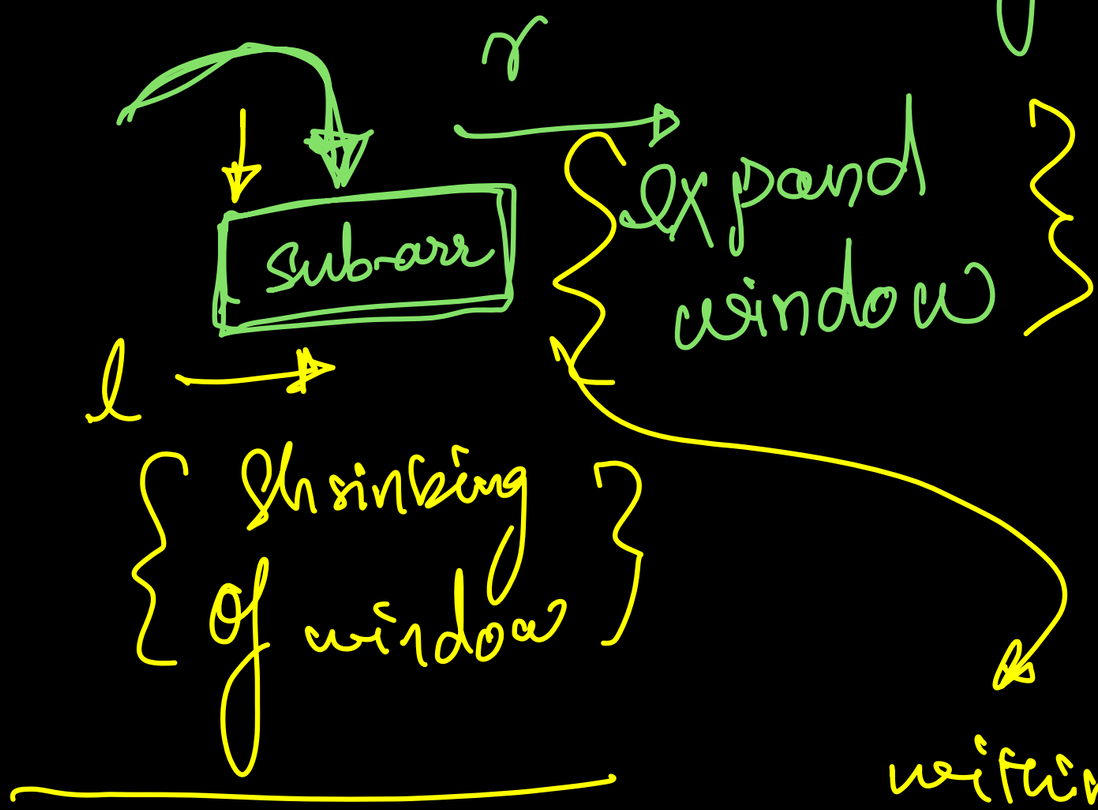
2) count no. of 0s in each subarray.

3) the subarray with 0's <= K

→ maintain the max size among these

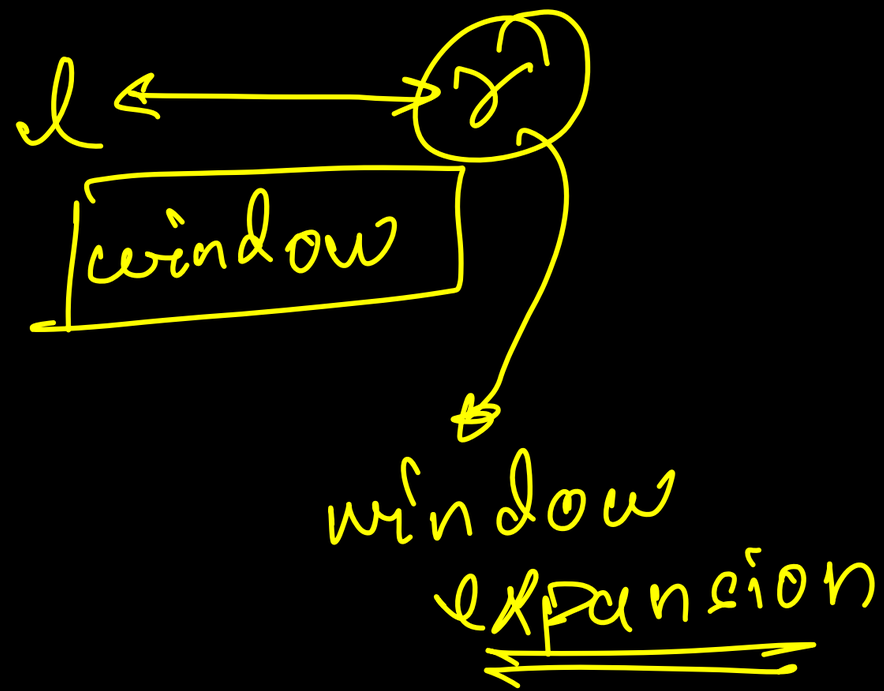
$O(n^2)$

soln 2 : Optimised code : Sliding window appr



Code logic:

```
int l=0, r=0
int zero = 0
while (r < n)
{
    if (a(r) == 0)
        zero++
    while (zero > k)
    {
        if (a(l) == 0)
            zero--
        l++
    }
    r++
}
```



{Shrink the window}

l++

}

[2 5] 2345

5 - 2 + 1 = 4

keep shrinking until cond of zero is within or = K

window_len = (r - l + 1)

ans = max (ans, window_len)

ends

O(n)

r++ // expand

Interview Question:

→ arr[n] → array

1 operation ⇒ a[i] pick and in case remaining (n-1) elems by 1.

Target :- make all n elems equal.

→ what are min no. of operation?

eg:- a:- [4, 3, 4]

steps :- 2 ✓

Soln:- pick ind: 2, incr others by 1

Step1:- 4 3 4 ⇒ 5 4 4

Step2:- pick ind 0 : 5, and incr others by 1

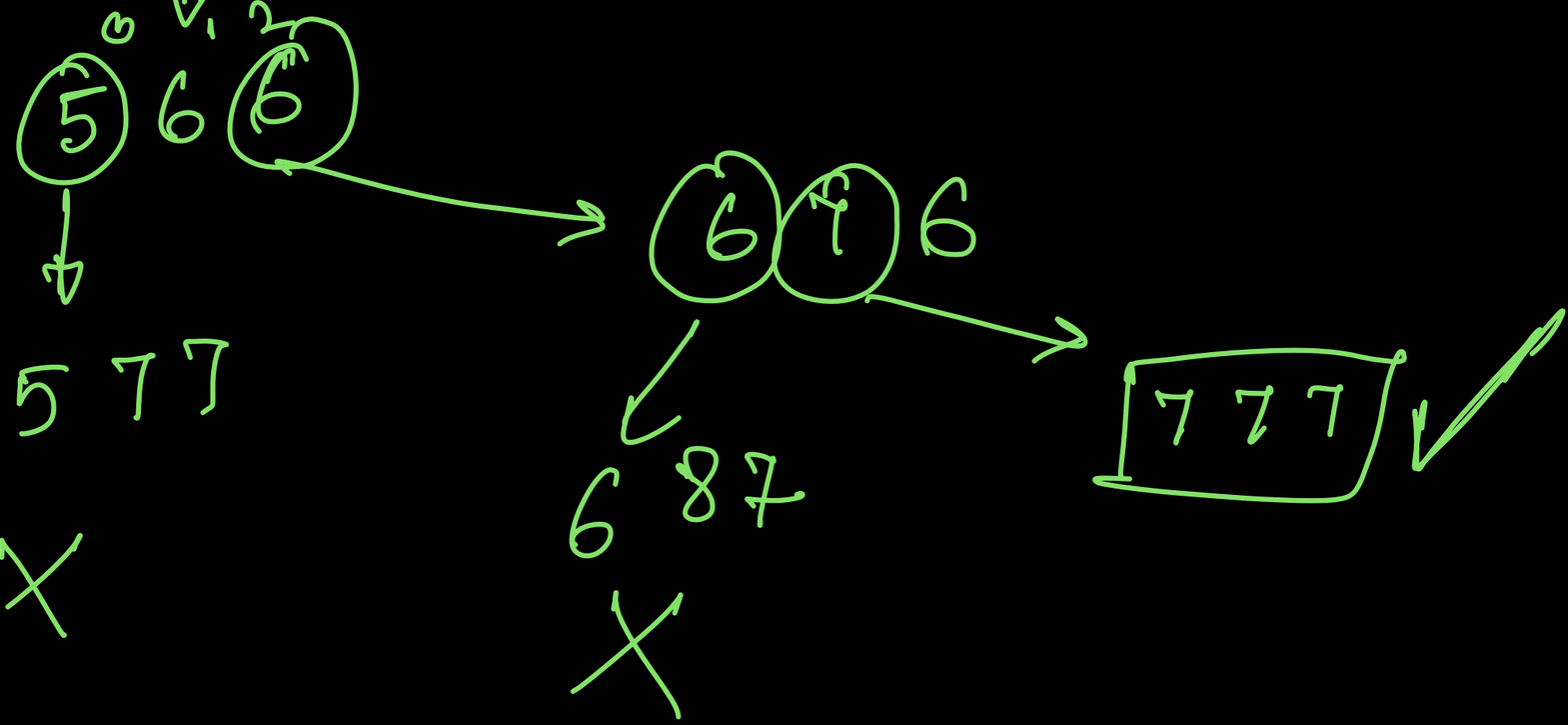
5 5 5

eg2 :- a = [4, 6, 5]

0 1 2

→ Step 1 : 4 6 5

ind 1 → pick / incr remaing.

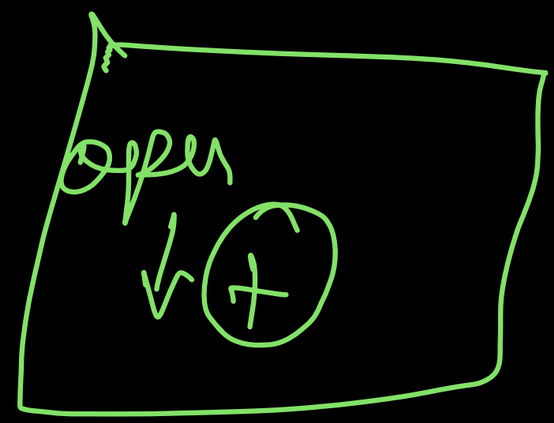


Logic :- if we pick min elem,

the distance/gap will only increment

gap ↑ oper ↑

gap ↓ oper ↓



HW : Think how to code ?

Dinner break

28th March → 2 sessions → 4PM, 8PM

29th march → 2 " → 4PM, 8PM

30th march → 2 " → 4PM, 8PM

Happy Learning

THANK - YOU