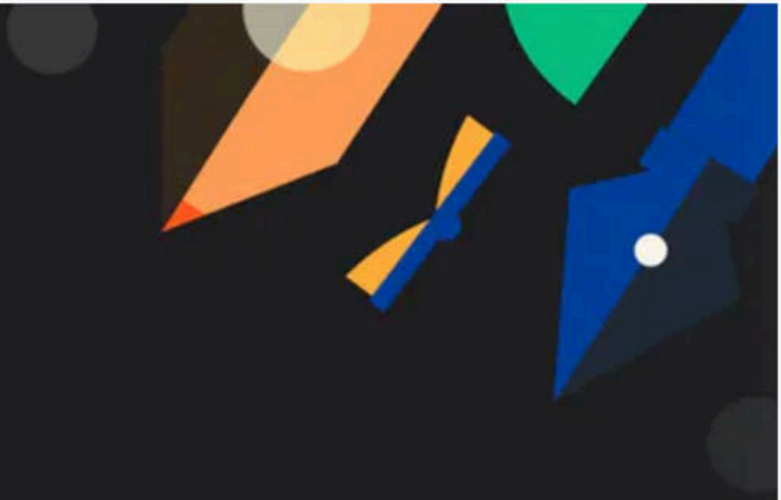
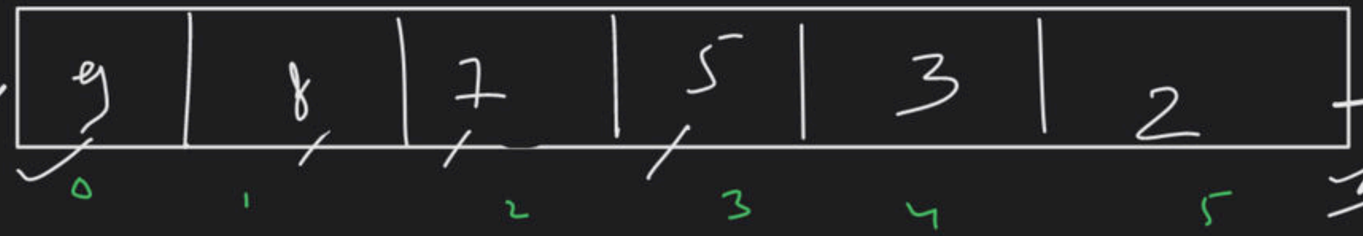


Queue - II

Foundation Course on Data Structures & Algorithms - Part II

Love Babbar • Lesson 5 • July 5, 2022

Q-1



Reverse Queue -> ?

stack

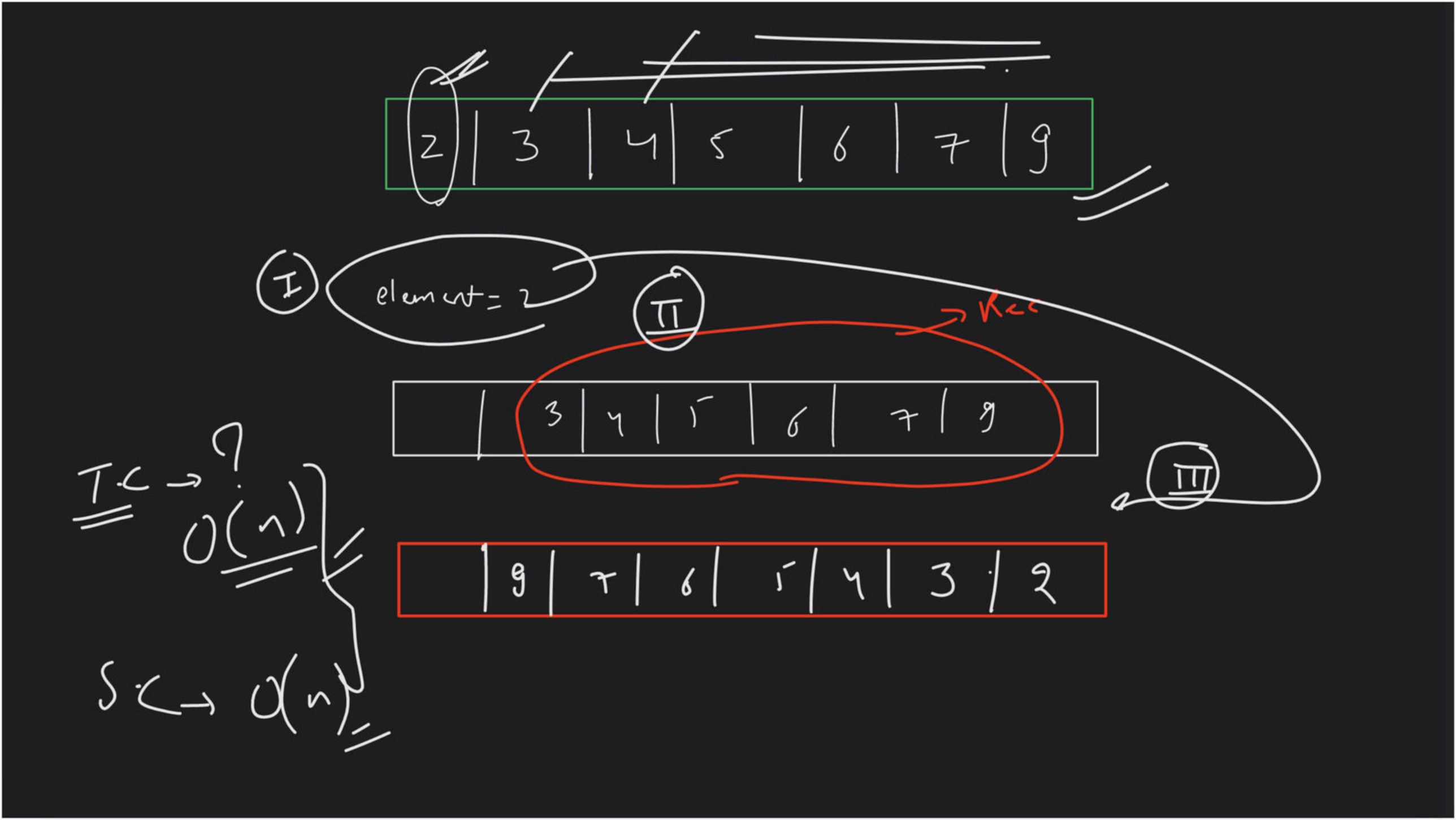
approach :-

(1) Queue -> stack -> O(n)

(2) stack -> queue -> O(n)

T.C -> O(n)

S.C -> O(n)

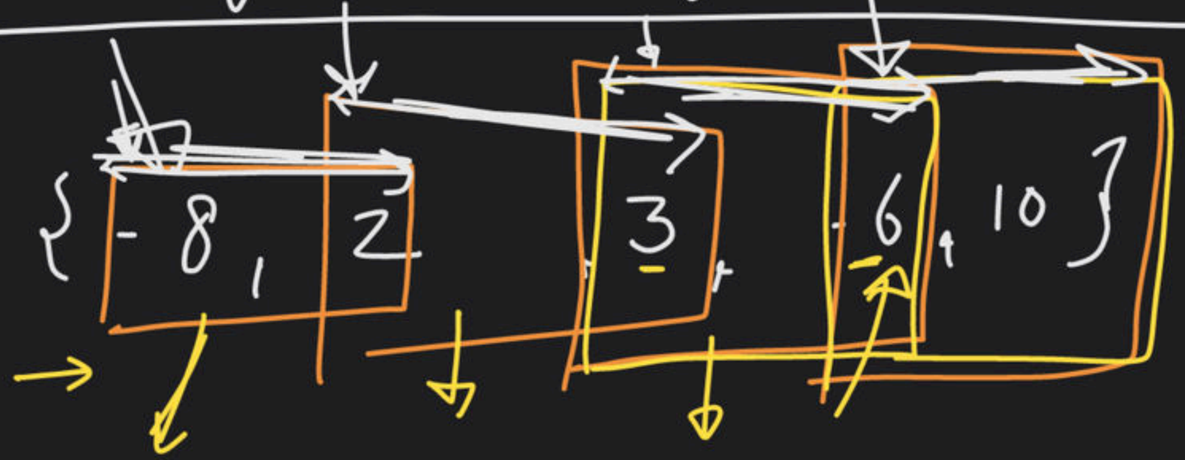

2 | 3 | 4 | 5 | 6 | 7 | 9
I
element = 2
II
Rec
3 | 4 | 5 | 6 | 7 | 9
III
T.C → ?
O(n)
9 | 7 | 6 | 5 | 4 | 3 | 2
S.C → O(n)

→ Recursion - ?

H/W

- implement Stack using Queue
- implement Queue using stack
- implement K stacks in an Array
- implement K Queues in an Array

→ ~~find~~ -ve integer in every window of size "K"



K=2

K=N

o/p → {-8, 0, -6, -6}

approach :- 1

```
for ( int i=0; i<n; i++ )
{
    for ( i<K times )
}
```

O(n*K)

T.C

K=2

{ -8 , 2 , 3 , -6 , 10 }

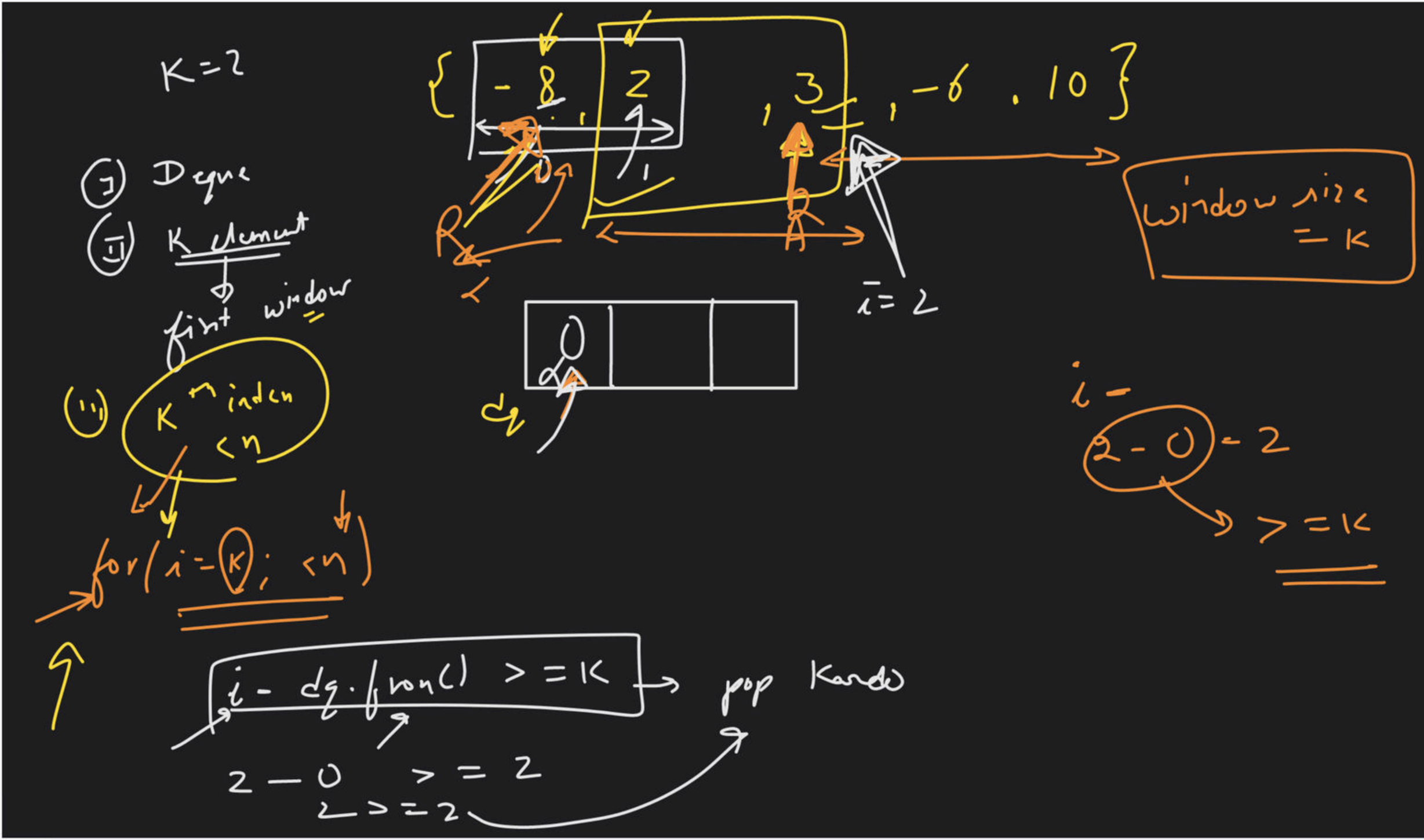
(i) Deque

(ii) K element

first window

(iii) K^th index < n

for(i = K ; i < n)



i - dq.front() >= K → pop karado

2 - 0 >= 2

2 >= 2

K = 2

{ -8, 2, 3, -6, 10 }

0 1 2 3 4

Rem

Add

i · i - dq.front >= K

4 - 3 >= 2

1 >= 2 → F

Pop

2 - 0 >= K

2 >= 2

(1) Dequeue

(2) find window

(3) save ans

(4) process remaining window

for (i = K; i < n)

{

// ans save

}

dq

arr →

{ 12 , -1 , -7 , 8 , -15 , 30 , 16 , 28 }

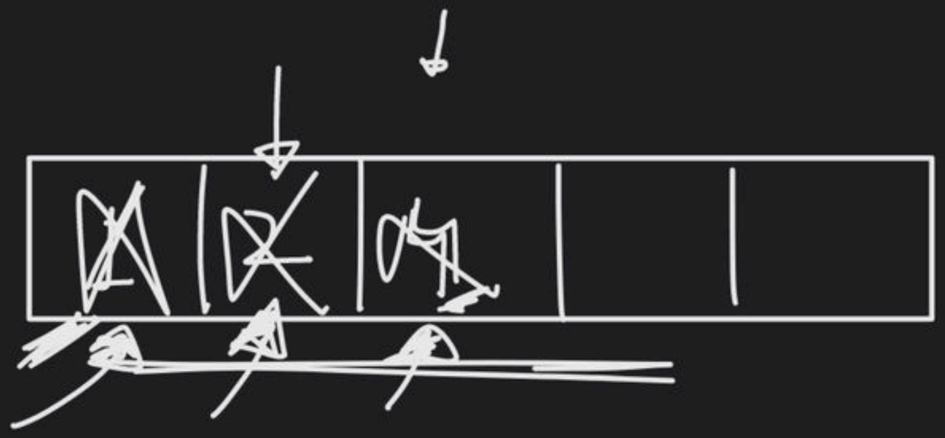
3 5 6 7

K = 3

3 - 1 >= 3

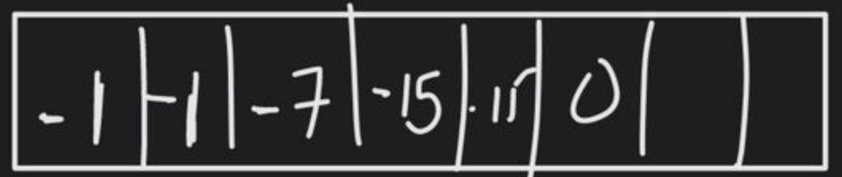
2 >= 3 - 1 F

→ each each



sahi

right or wrong



ans

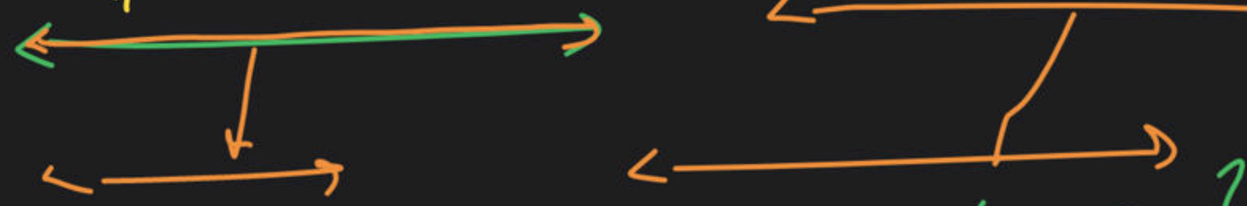
Sol 1

→ Optimal Solution → explore

→ Reverse first K element of Queue

= { 1, 2, 3, 4, 5, 6, 7 }

K = 3



o/p -> { 3 2 1 4 5 6 7 }

Algo:-

① I/P K elements → put in stack → O(K)

② K element → queue → O(K)

③ (n−K) element → queue → O(n−K)

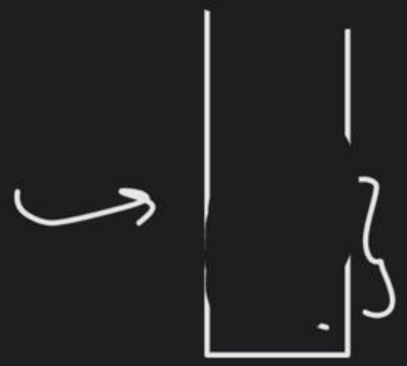
T.C → O(n)

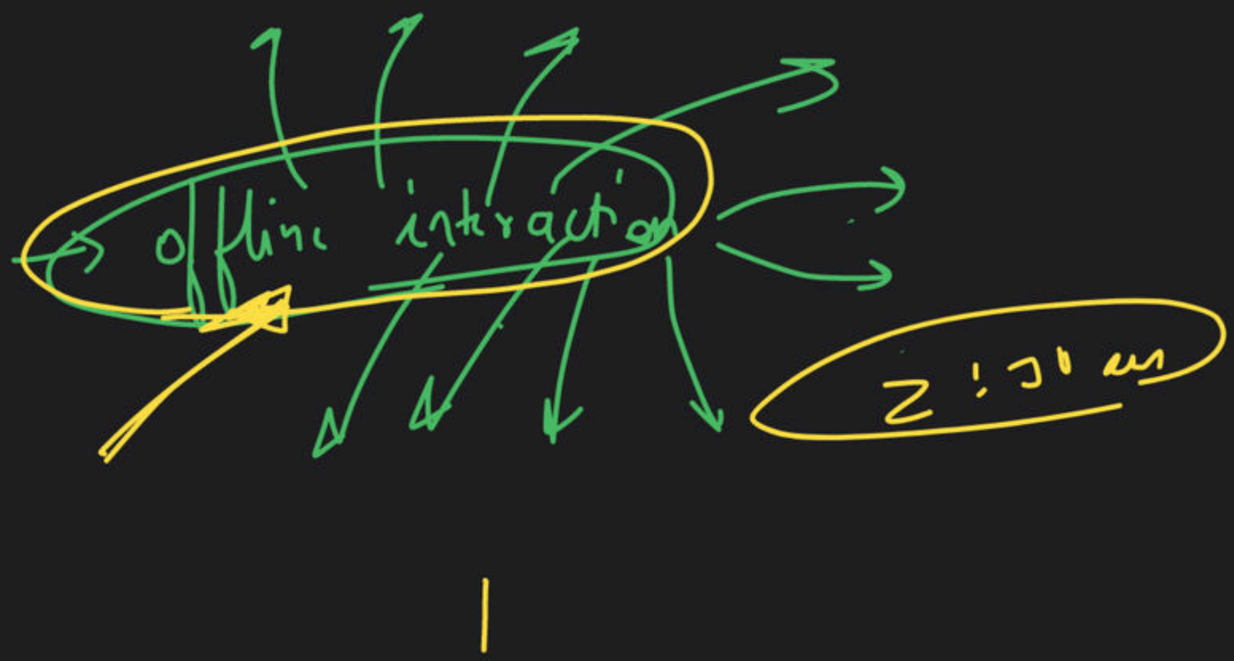
n = 6

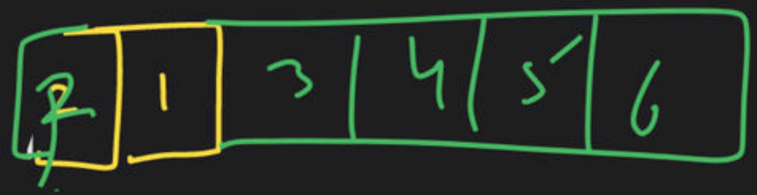
K = 2

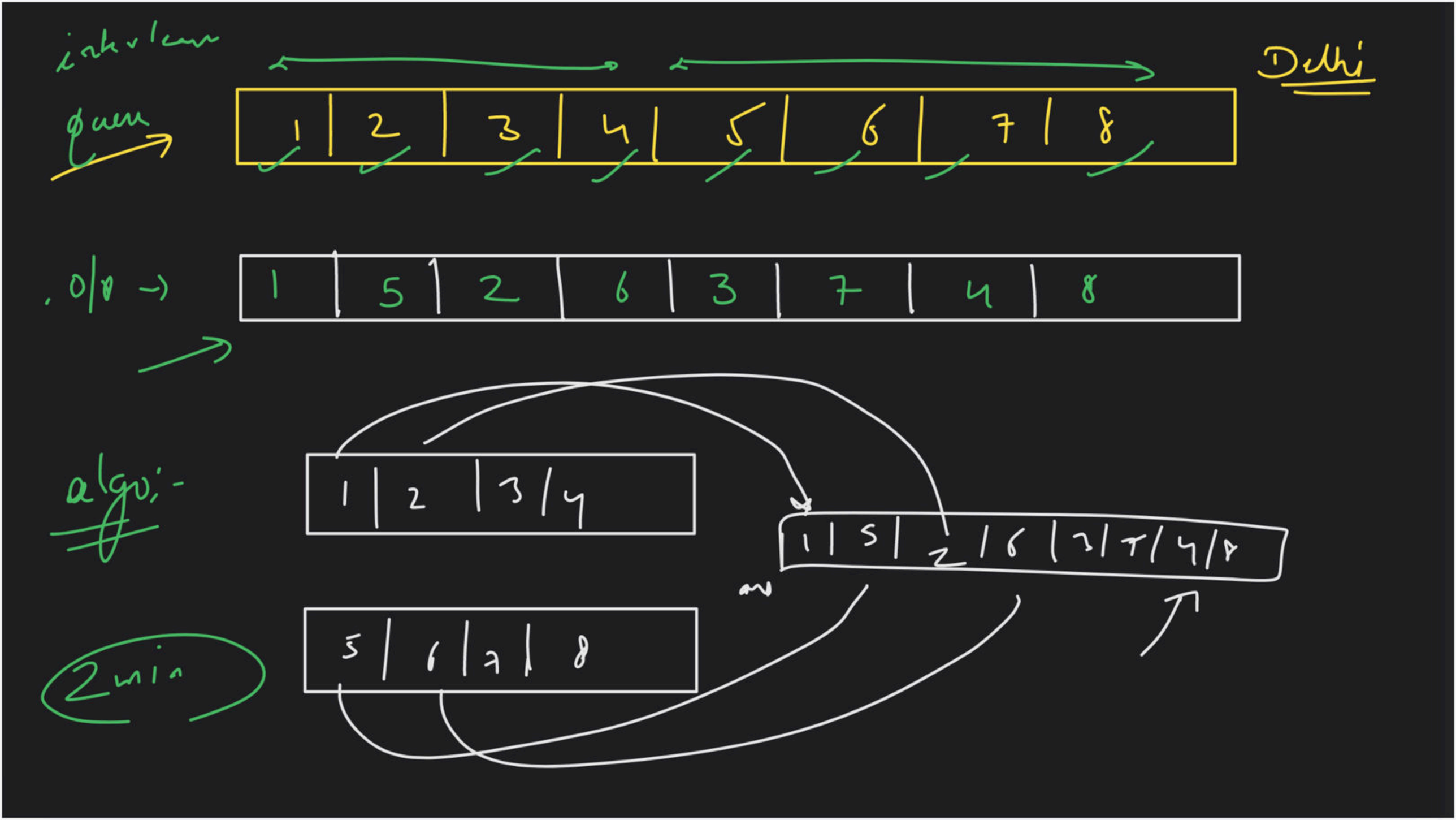
n − K = 4

S.C → O(K)




offline interaction
2:30 ms


2
1
3
4
5
6


interview
Delhi
Ques
1 2 3 4 5 6 7 8
O/P →
1 5 2 6 3 7 4 8
algo:-
1 2 3 4
5 6 7 8
1 5 2 6 3 7 4 8
2 min

Cond^{n} → Stack

T.C → O(n)
S.C → O(n)

(I) pur first half in stack

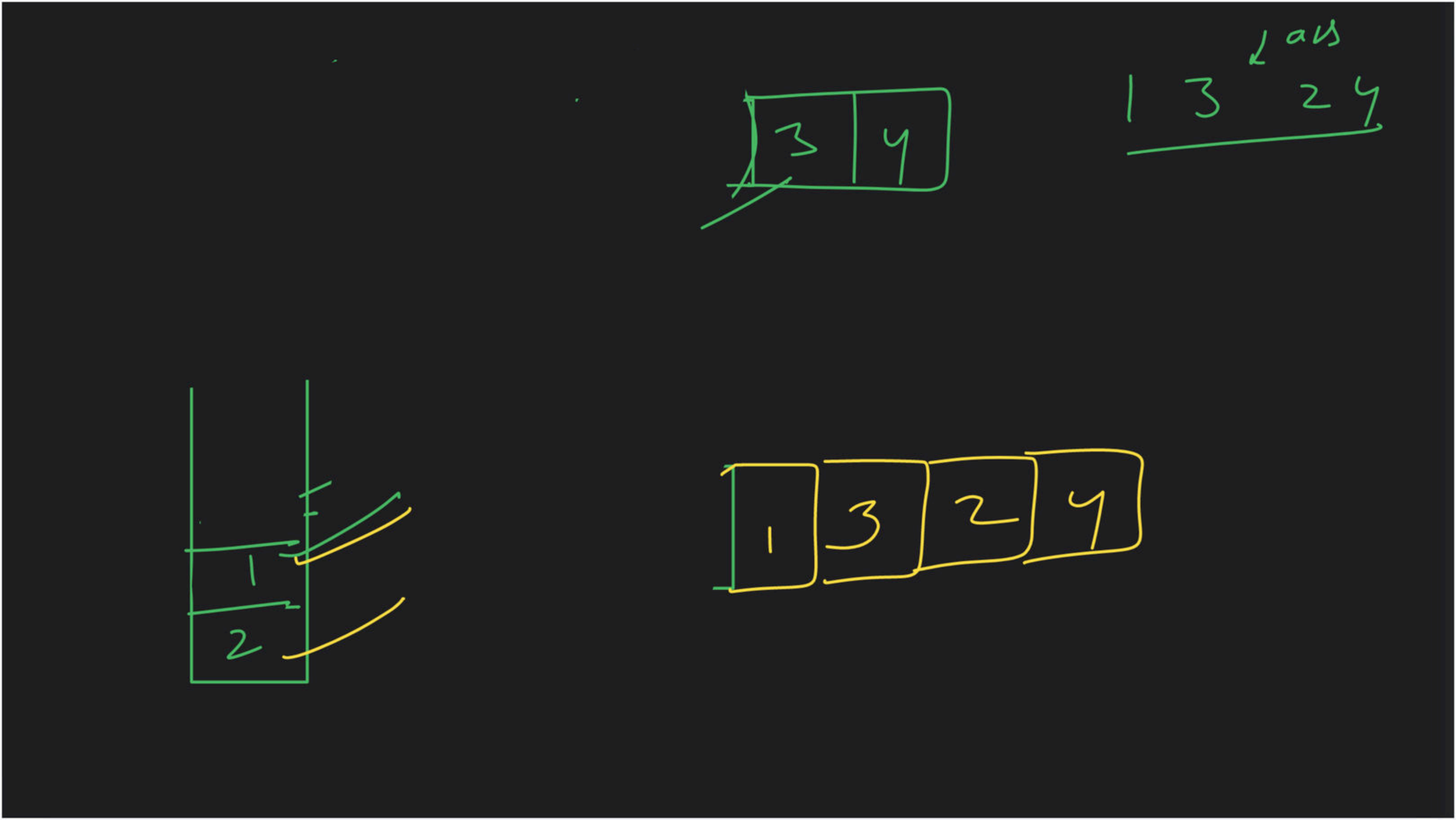
(II) stack → Queue

(III) first half → queue me push

(IV) put first half in stack

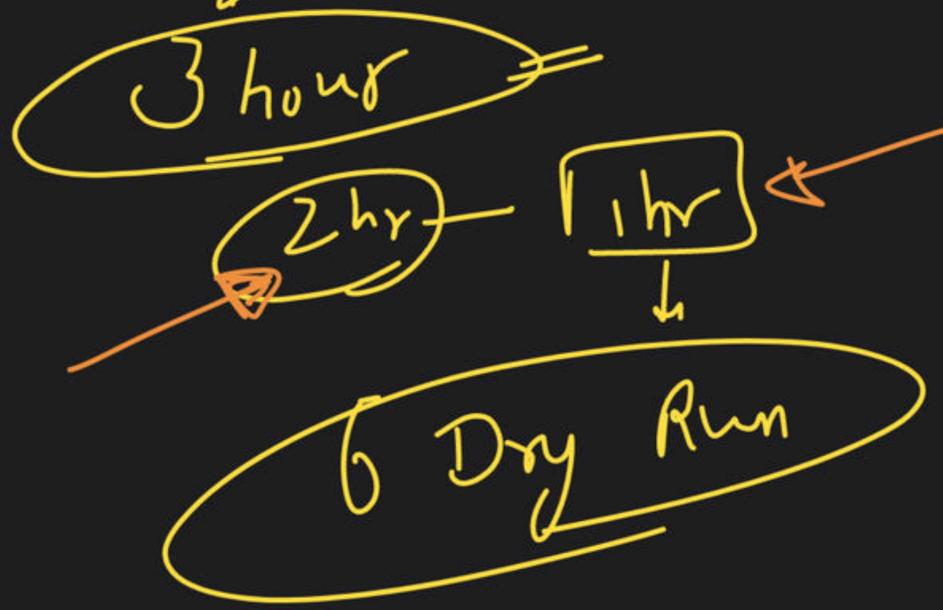
(V) merge stack & queue

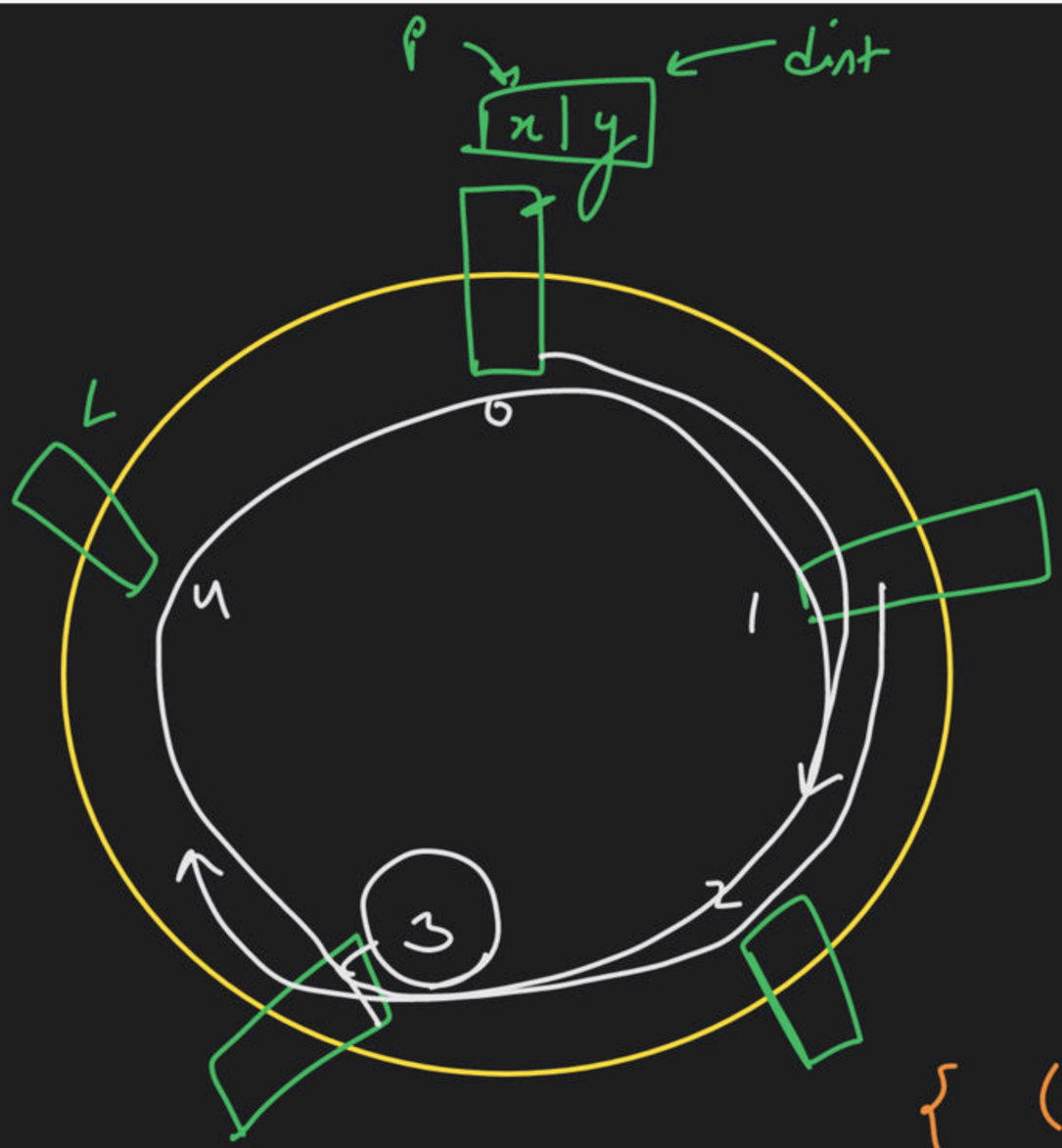

```
while (!s.empty())
{
q.push(s.top())
s.pop()
q.push(q.front())
q.pop()
```


Ans
1 3 2 4
3 4
1
2
1 3 2 4

Queue →

Circular Tour / Gas Station / Petrol Pump





find the

Circular tour

that visits all petrol pump

{ (6,4) , (3,1) , (7,4) }

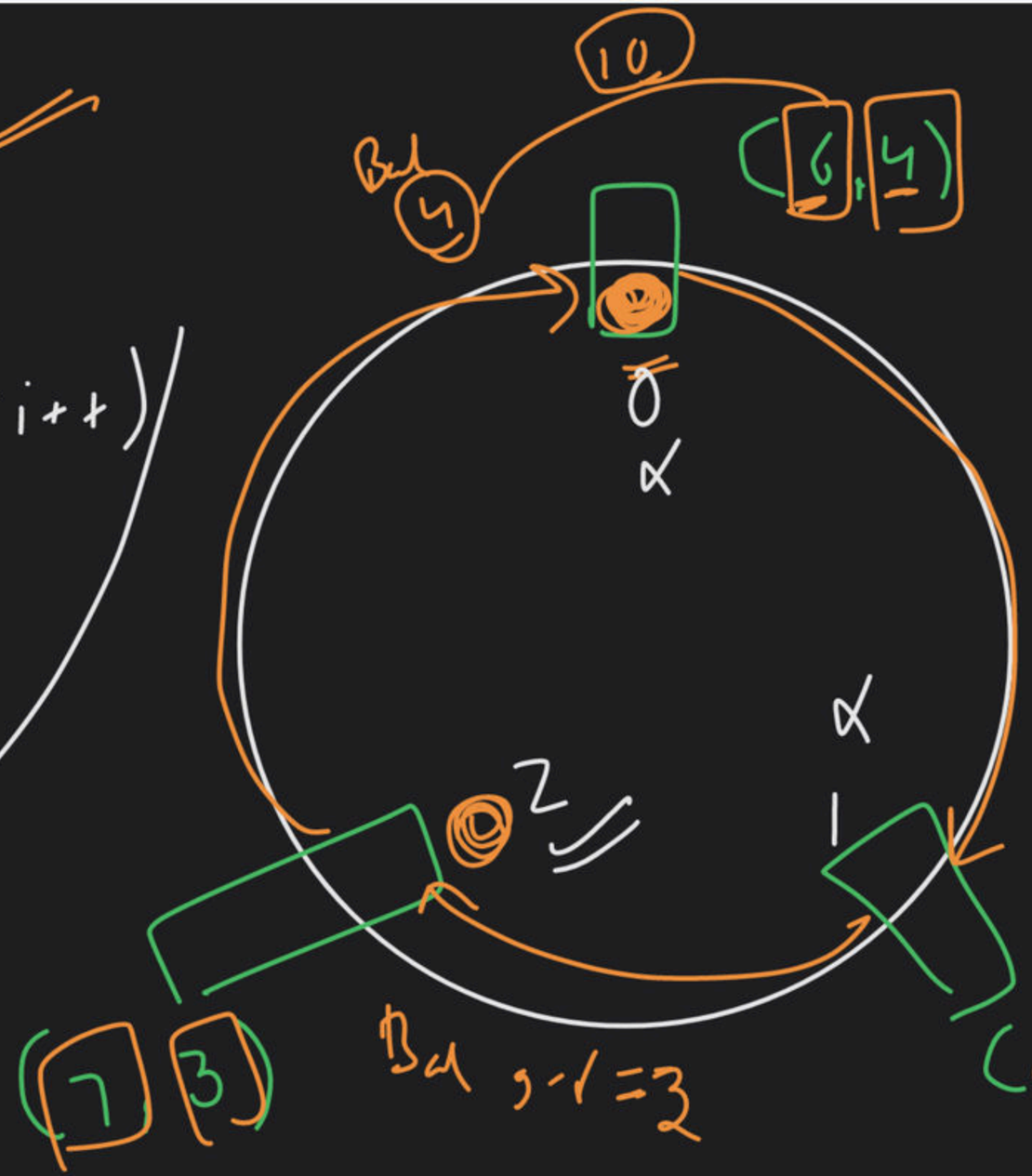
L

YT

B.F #1

for(i=0; i<n; i++)
{
// transfer
}

O(n²)



#2 → Queue

ans = 2

#3 → Variable

declare below

Bal = 10-4 = 6

→ Sum of max & min elements of all subarrays of size "K".

↓

HEAP

arr[] = { 2, 5, -1, 7, -3, -1, -2 }

K = 4

① deque → max → decr inc → K

② deque → min

1st → [2, 5, -1, 7] → max = 7, min = -1 → 6

2nd → [5, -1, 7, -3] → max = 7, min = -3 → 4

3rd → [-1, 7, -3, -1] → max → 7, min = -3 → 4

[7 -3 -1 -2] → max = 7 min = -3

Sum = 6 + 4 + 4 + 7

= 18 → o/p