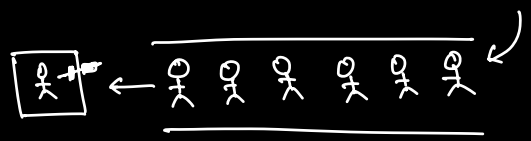
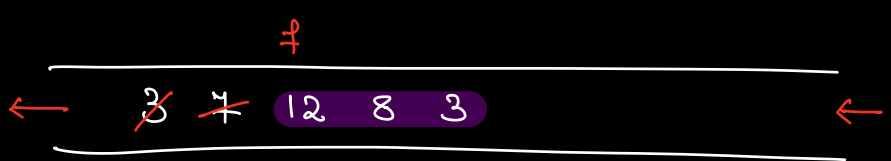
QUEUE :-



Queue → ADT

- enqueue(x) : insert
- dequeue() : delete.
- front()
- isEmpty()
- size()

Quiz eq(3), eq(7), eq(12), dq(), dq(), eq(8), eq(3)



Quiz

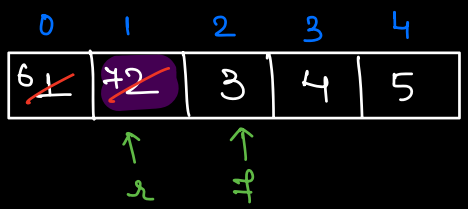
eq(4), dq(), eq(9), eq(3), eq(7), eq(11), eq(20), dq()

Implement Queue :

1. Array

2 pointers
- front
- rear

```
int arr[5];
```



eq(1)
eq(2)
eq(3)
eq(4)
eq(5)
dq()
dq()
eq(6)
eq(7)

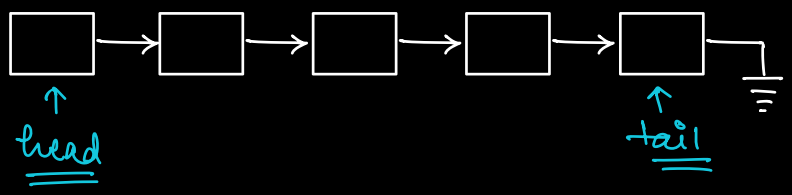
$(r+1) \longrightarrow (r+1)\%N$

$(4+1) \longrightarrow (4+1)\%5$

⟶ Size == N : Queue is full.

HW : Implement Circular Queue using Array.

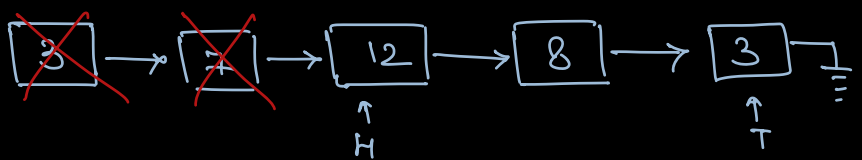
Linked List



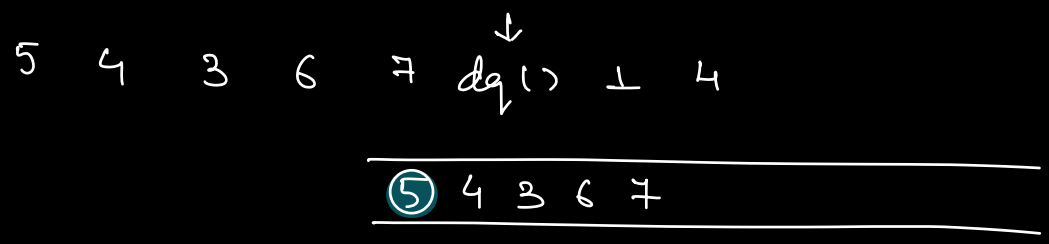
Insert : O(1)
Delete : O(1)

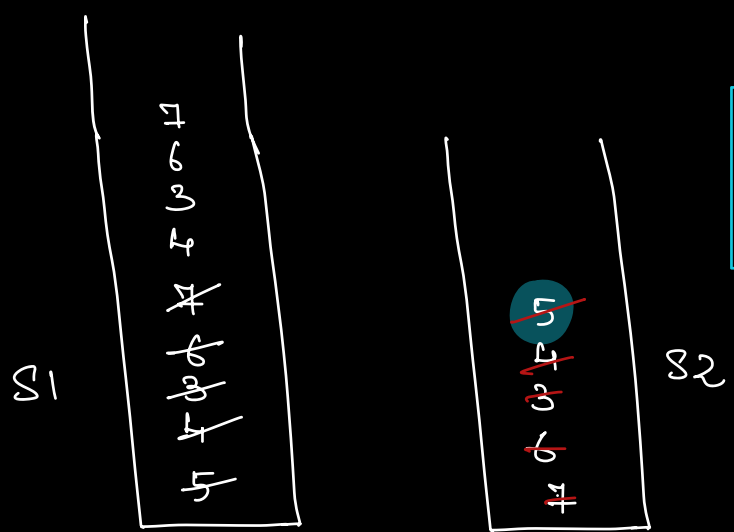
Insert : O(1)
Delete : O(N)

eq(3), eq(7), eq(12), dq(), dq(), eq(8), eq(3)



Q. Implement Queue using Stack.
- push(x)
- pop()

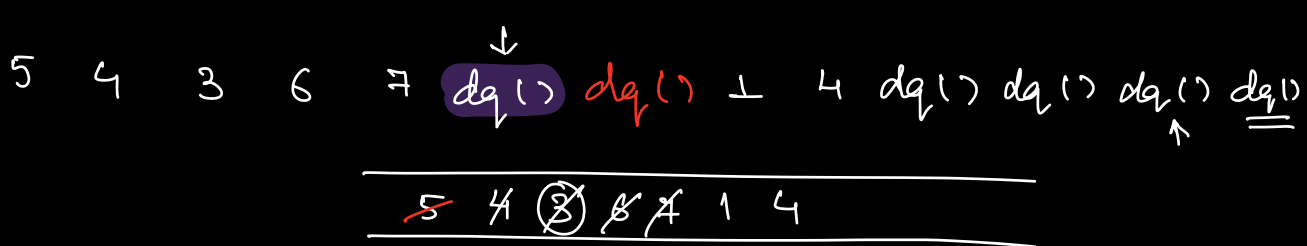


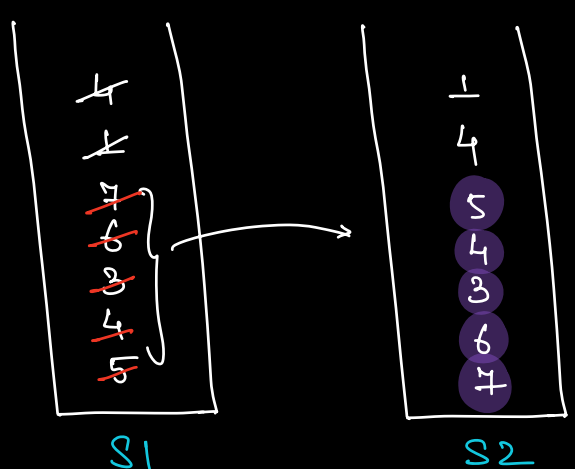


TC:

enqueue(x) : O(1)

dequeue() : O(N)





```
Void enqueue(x) {
    S1.push(x);
}
```

```
Void dequeue() {
    if (S2.isEmpty()) {
        while(!S1.isEmpty()){
            S2.push(S1.top())
            S1.pop();
        }
    }
    if (!S2.isEmpty())
        S2.pop();
}
```

TC: enqueue(x): O(1)
dequeue(): —

for 1st dequeue() → N iterations
for rest (N-1) dq() operations → (N-1) iterations

N dq() operations → N + (N-1) = 2N-1

1 dq() operation → $\frac{2N-1}{N}$

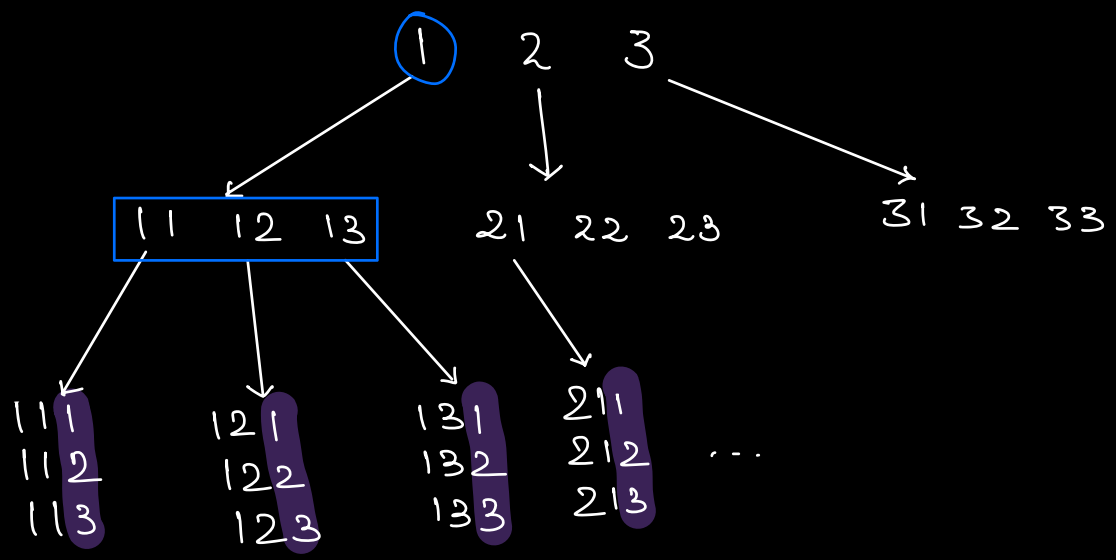
→ O(1)

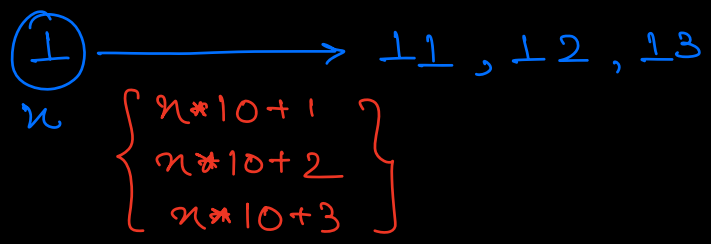
Amortized TC.

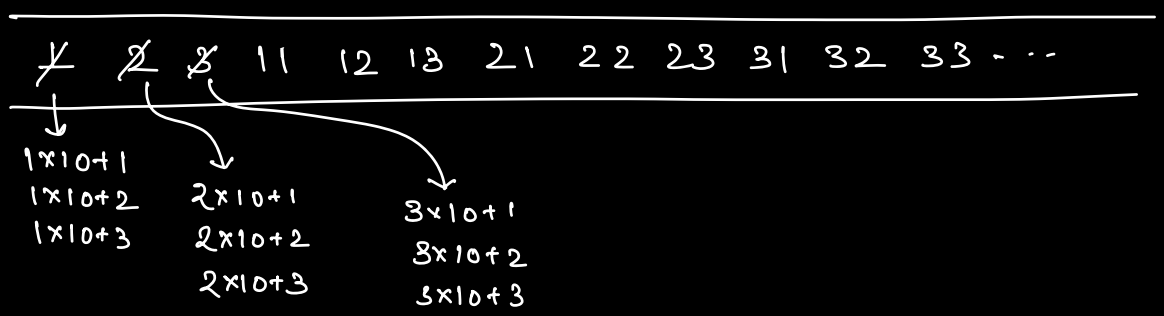
Q. N^{th} number using the digits 1, 2 OR 3.

(No digit apart from 1, 2 and 3 is allowed).

1 → N = 1
2 → N = 2
3 → N = 3
11 → N = 4
12 → N = 5
13 → N = 6
21 → N = 7
22 → N = 8
23
31
32
33
111
112
113
⋮







BFS ≡ Queue

HW: Implement

TC : O(N)

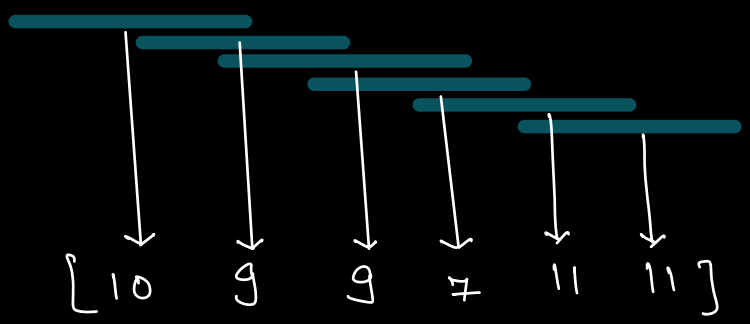
SC : O(N)

Q. **Sliding Window Maximum.**

* Given an Array of size N, find the MAX of every window of size = K.

Google
FB
MS
Amazon
Adobe
DEShaw
Arcesium.
...

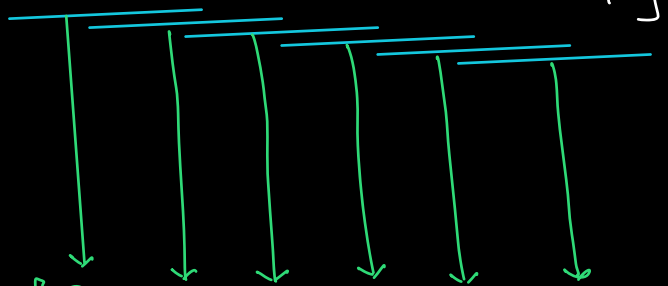
A: 10 8 9 7 6 5 11 3 K = 3



[10 9 9 7 11 11]

Quiz

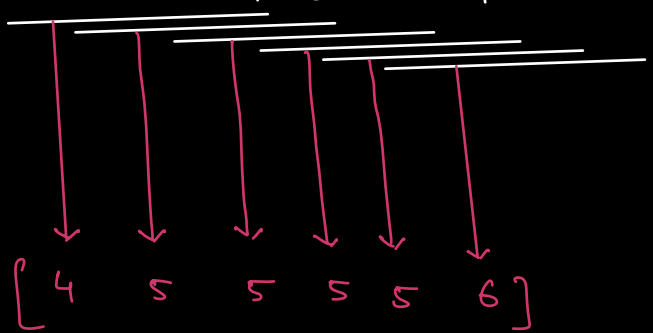
A: [1 3 -1 -3 5 3 6 7]



[3 3 5 5 6 7]

Quiz

A: [3 2 3 4 5 5 4 5 6]

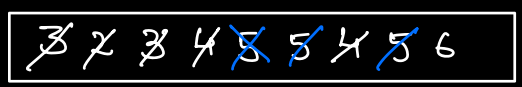


[4 5 5 5 5 6]

Brute force : ✓

A : [3 2 3 4 5 5 4 5 6] K=4

Box



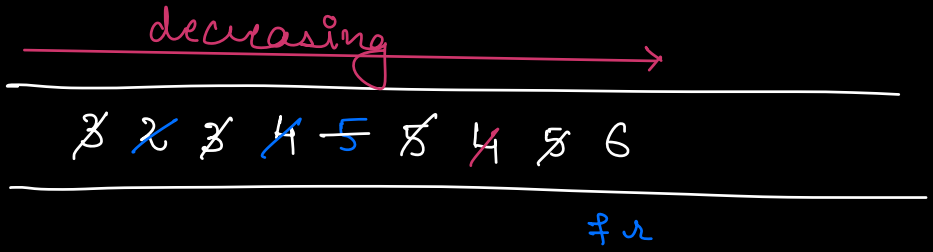
Dequeue

(Doubly Ended Queue)

Ans : [4, 5, 5, 5, 5, 6]

A : [3 2 3 4 5 5 4 5 6]

DQ



ans : [4 5 5 5 5 6]

=

A: 10 1 8 9 7 6 5 11 3 k=3.

DQ

10 1 8 9 7 6 5 11 3

f r

ans: [10, 9, 9, 9, 7, 11, 11]

Dequeue :

- push_back(x) → front()
- pop_back() → rear()
- push_front(x)
- pop_front()
- isEmpty()
- Size()
- :

⇒ Doubly linked list:

{TC: O(N)
SC: O(K) → O(N)}

HW

*