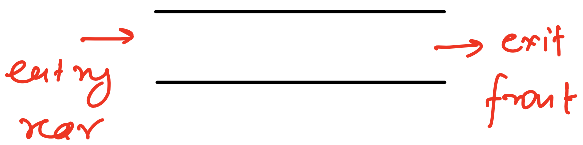
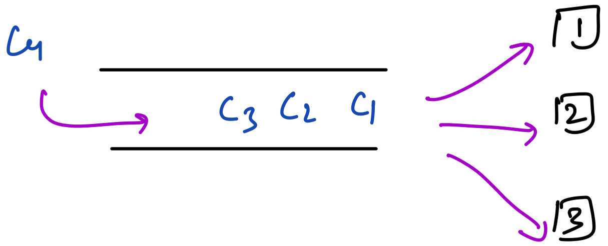
Queues: Implementation & Problems

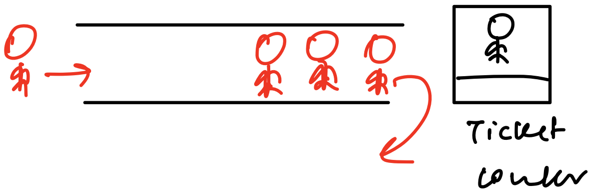
Queue



first In first Out
(FIFO)

Customer Care





Operation

1. Enqueue(x) → Insert x from rear end
2. Dequeue() → Remove data from front end
3. isEmpty() → check if queue is empty
4. front() → get the data at the front end
5. Rear() → get the data at rear end

$TC = O(1)$

Ques → Implement queue using dynamic array.

enqueue(3)

enqueue(5)

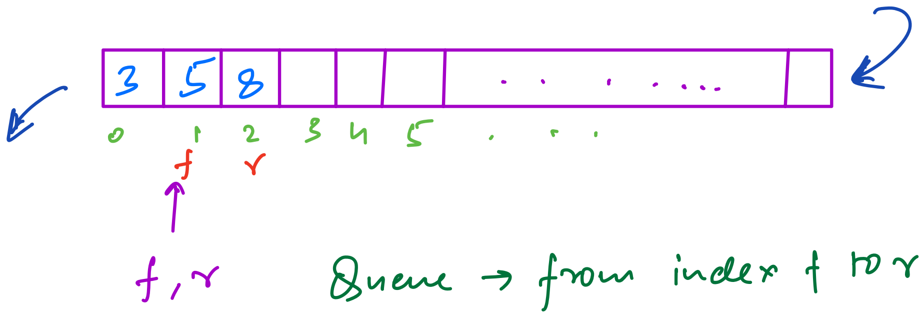
enqueue(8)

dequeue()

isEmpty() → false

front() → 5

rear() → 8



Queue → from index f to r

[f, r] subarray

f = ~~0~~ 1

r = ~~-1~~ ~~0~~ ~~1~~ 2

```
void enqueue(x) {
    r++
    A[r] = x
}
```

//Overflow → use dynamic array

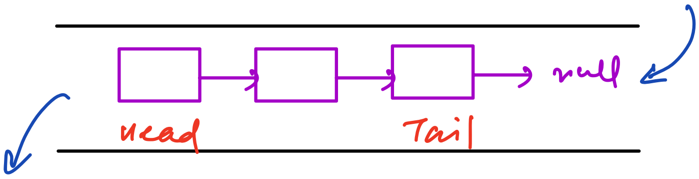
```
int dequeue() {
    if(isEmpty()) return -1
    f++
    return A[f-1]
}
```

```
bool isEmpty() {
    return f > r
}
```

```
int front() {
    if(isEmpty()) return -1
    return A[f]
}
```

```
int rear() {
    if( isEmpty()) return -1
    return A[r]
}
```

Ques → Implement Queue using linked list



1. enqueue(x) →

   insert at Tail

2. dequeue() →

   remove from Head

3. isEmpty() → (Head == null)

4. front() → Head.data

5. rear() → Tail.data

TC	Insertion	Deletion
Head	O(1)	O(1) ✓
Tail	O(1) ✓	O(N)

TC = O(1) ∀ operations

enqueue(3) enqueue(7) enqueue(12) dequeue()

dequeue() enqueue(8) enqueue(3)

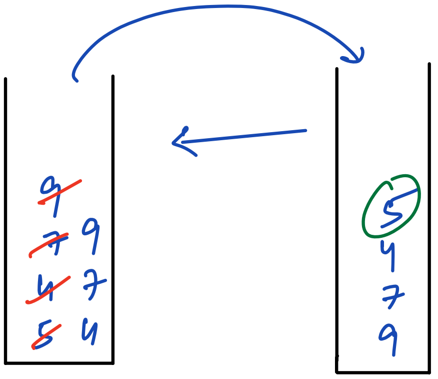
~~3~~ ~~7~~ 12 8 3

enqueue(4) dequeue() enqueue(9) enqueue(3) enqueue(7)
enqueue(11) enqueue(20) dequeue()

~~4~~ ~~9~~ 3 7 11 20

Ques → Implement queue using Stack.

[5 4 7 9 De 8 10 De De]



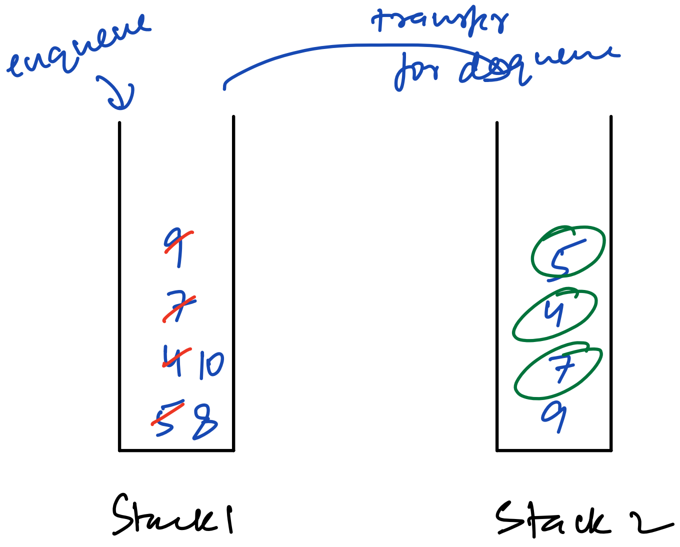
Enqueue → O(1)

- Push x in Stack 1

Dequeue → O(N)

- Transfer all from Stack 1 to Stack 2
- Remove top element in Stack 2
- Transfer all back from Stack 2 to Stack 1

[5 4 7 9 De 8 10 De De]



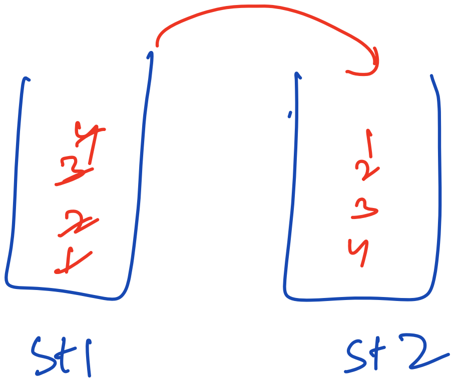
```
void enqueue(x) {
    st1.push(x)
}

int dequeue() {
    if (isEmpty())
        return -1
    if(st2.isEmpty()) {
        move();
    }
    return st2.pop()
}
```

```
void move() {
    while(! st1.isEmpty()){
        st2.push( st1.pop())
    }
}

bool isEmpty() {
    return (st1.isEmpty()
        && st2.isEmpty())
}
```

If TC of move() = O(K) ⇒ next K dequeue() operation will have TC = O(1)



dequeue() → 4

dequeue() → 1

dequeue() → 1

dequeue() → 1

$$\text{avg.} = \frac{4+1+1+1}{4} = \frac{7}{4} \approx 2$$

So, avg. TC of dequeue = O(1)

↓

amortized

Question

Given an integer N, find N^{th} number that can be formed by digits 1 & 2 only.

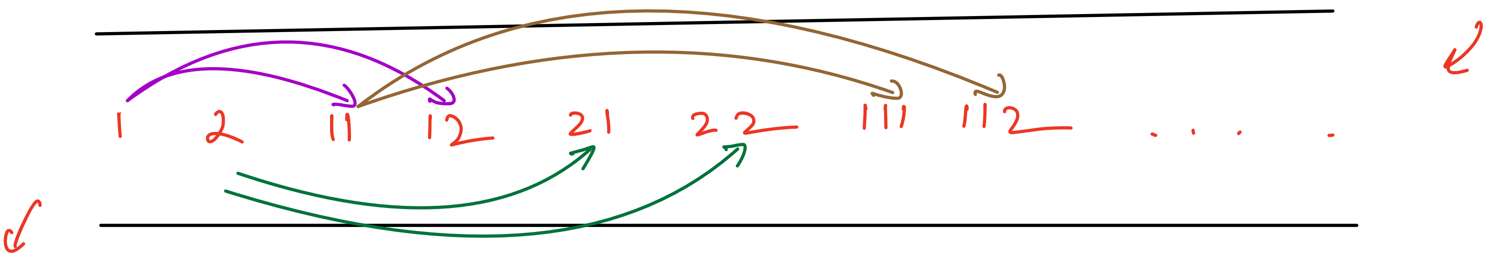
1	2	11	12	21	22	111	112	121	122	. . . - - ,
N = 1	2	3	4	5	6	7	8	9	10	. . . "

0	10	20
1	11	21
2	12	22
3	13	23
⋮	⋮	⋮
9	19	29

1 2

11 12 21 22

111 112 121 122 211 212 221 222



$x \rightarrow x*10+1$

$x*10+2$

Code

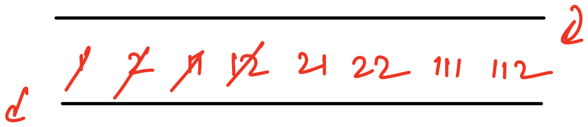
```
if (N<=2)    return N
q.enqueue(1)
q.enqueue(2)
i=3
while(i<=N) {
    x= q.dequeue()
    a= 10*x +1
    b = 10*x +2

    if(i==N)    return a
    if(i+1 ==N)    return b
    q.enqueue(a)
    q.enqueue(b)

    i+=2
}
```

$TC = O(N)$

$SC = O(N)$



$N=10$

i= ~~3~~ ~~5~~ ~~7~~ 9

x= ~~1~~ ~~2~~ ~~11~~ 12   b=122

HW → find N^{th} number using only prime digits ? (2,3,5,7)

Double Ended Queue (Deque)

→ enqueue & dequeue from both sides

1. enqueueFront(x)
2. enqueue Rear(x)
3. dequeue front()
4. dequeue Rear()
5. isEmpty()

Implement Deque → doubly linked list

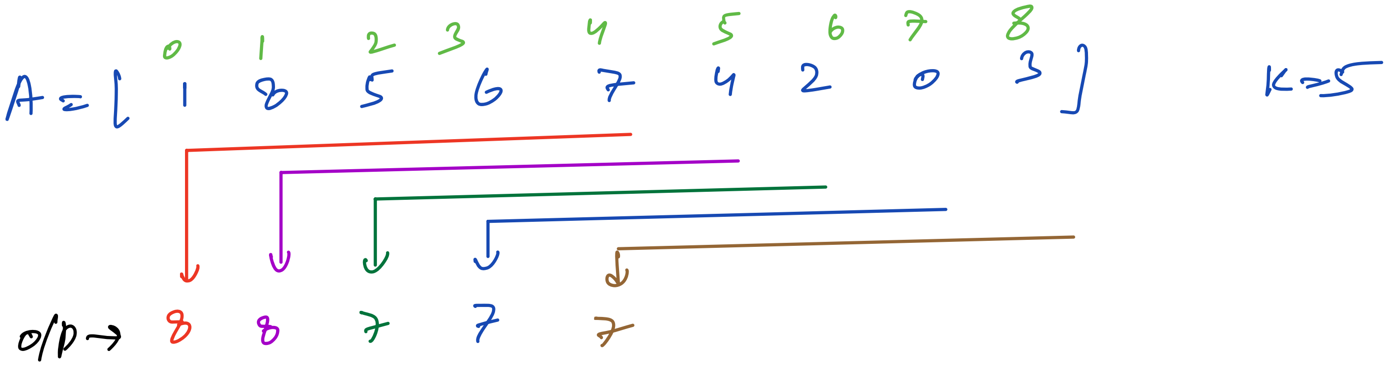
TC = O(1) ∀ operations

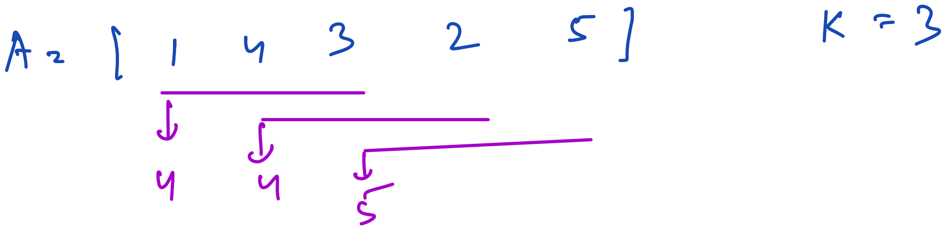
Question

Given an integer array & an integer K.

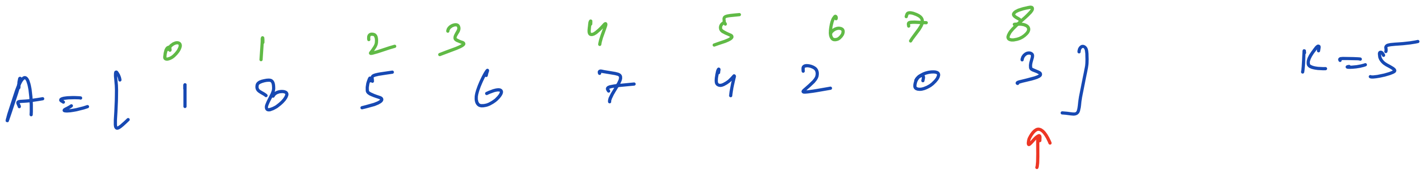
find the max element of subarrays of size K.

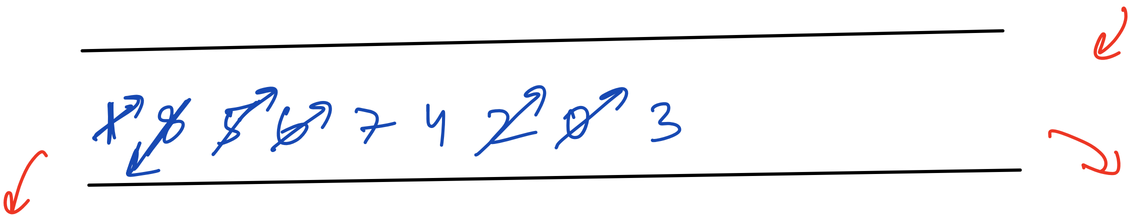
sliding window





[4 4 5]





o/p → 8 8 7 7 7

store ~~data~~ → index of data

Code

```
for(i=0 to K-1){
    while(! q.isEmpty() && A[q.rear()] <= A[i]){
        q.dequeueRear()
    }
    q.enqueueRear(i)
}
print( A[q.front()] )    ← max in 1st window

for(i=K to n-1){
    while(! q.isEmpty() && A[q.rear()] <= A[i]){
        q.dequeueRear()
    }
    q.enqueueRear(i)
    if( q.front() == i-K){    //out of window
        q.dequeueFront()
    }
    print( A[q.front()])
}
```

TC = O(N)

SC = O(K)

$$A = [\begin{array}{ccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 8 & 5 & 6 & 7 & 4 & 2 & 0 & 3 \\ & & & & & & & & \uparrow \end{array}]$$

K = 5

~~0~~ ~~1~~ ~~2~~ ~~3~~ 4 5 ~~6~~ ~~7~~ 8

o/p → 8 8 7 7 7

Scenario: Real Time Stock Trading Alerts

A = [220 215 230 225 240 235 230 245 250]