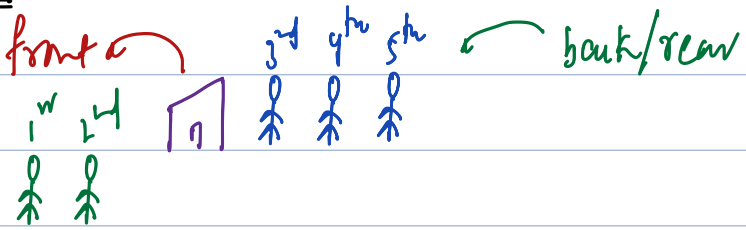
Todays Content

1. Queue Intro
2. Implement Queue using Stack

Queue



Queue is datastructure, where you enter at back/rear & exit at front()
Property: FIFO: First In First Out

Functions:

Enque(x) :	Insert x at rear/back end at enque
deque() :	delete ele at front end
front() :	Return ele at front end
size() :	Return no: of ele in queue.

Examples:

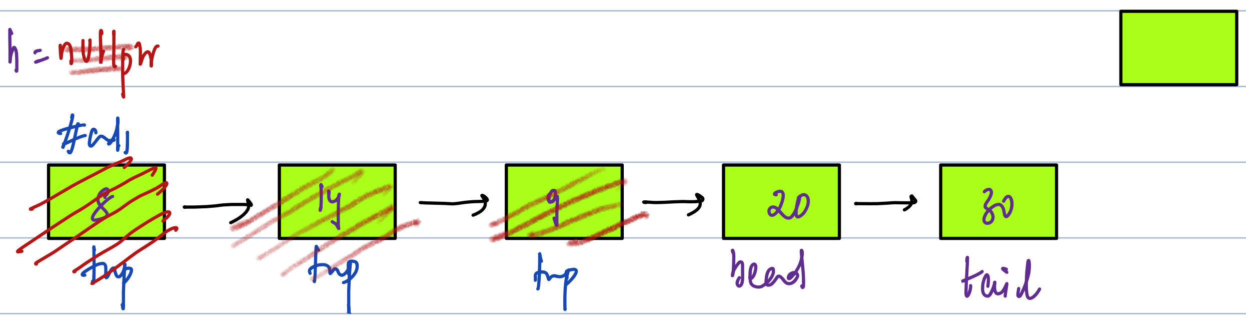
						14		9	9		
8	14	9	20	↑	30	front()	↑	front()	front()	↑	60
				8			14			9	

front　　　　　　　　　　　　　rear

~~8~~ ~~14~~ ~~9~~ 20 30 60

Implementation: Using arrays : TODO

Implementation: Linked List



8	14	9	20	↑	30	front()	↑	↑	60
✓	✓	✓	✓	8	✓	14	14	9	

front() front()

Code

```
class Node{
    public:
        int data;
        Node* next;
        Node(int x){
            data=x, next=nullptr;
        }
}

Node *h = nullptr, *t = nullptr;
int c=0;
void esque(int x){
    Node *nn = new Node(x);
    if( h==nullptr){
        h=nn;
    }   t=nn;
    else{
        t->next=nn;
    }   t=nn;
}   c++;


void deque( ){
    if( h==null){
    }   return;
    Node *tmp = h;
    h = h->next;
    delete tmp;
}   c--;
```

```
int front() {
    if( h==null) {
        return -1; #depends on question.
    }
    return h->data;
}

int size() {
    retrn c;
}
```

C++

queue<type> que;

que.push( )	Insert x at rear/back end at enque
que.pop()	delete ele at front end
que.front()	Return ele at front end
que.back()	Return ele at back end.
que.size()	Return no: of ele in queue.

Java

Queue<Type> que = new LinkedList<>();

que.add(x) :	add ele at rear/back end
que.poll() :	Remove & return element at front end
que.peek() :	Return element at front end
que.size() :	Return size()

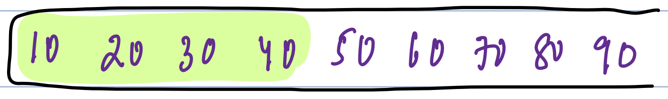
Python

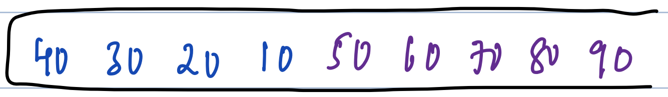
que = deque();

que.append(10);	add ele at rear/back end
que.popleft();	Remove & return element at front end
que[0];	Return element at front end
len(que);	Return size()

18 Given Que, Reverse First k elements.

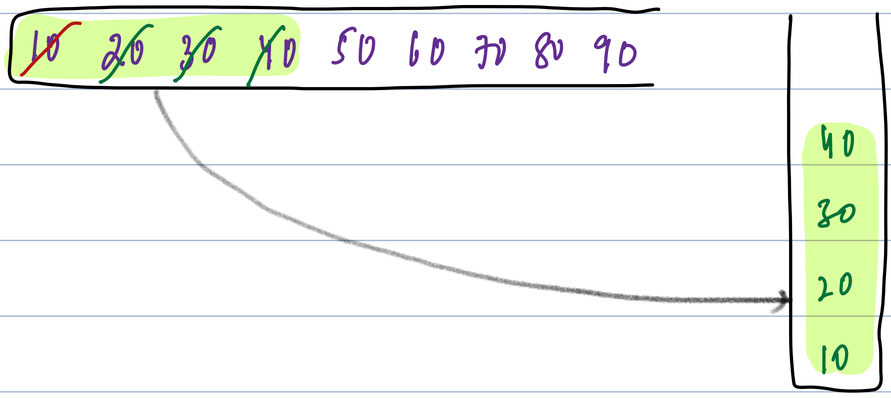
k = 4



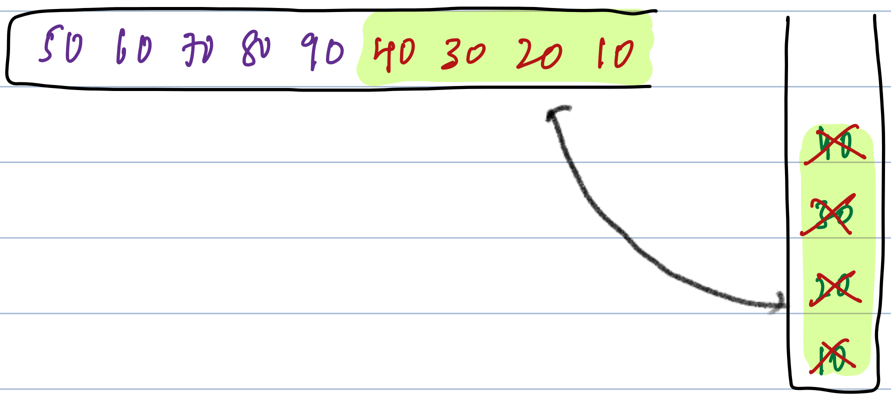


Idea 1:

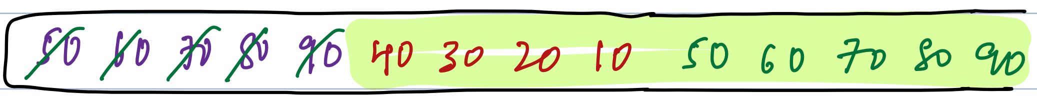
Step1: Deque k ele from queue & insert in stack



Step2: Delete k elements from stack & insert in Queue



Step3: For N-k Times
Insert front element in Queue & delete it



20 Implement Queue using Stacks

Queue Operations:		Stack Operations
enque( )	Every queue function should be implemented using stack function only	push( )
deque( )		pop( )
front( )		peek( )
size( )		size( )

Operations:

5 4 7 9 ↑ 8 f( ) 10 ↑ ↑ 14 ↑ ↑ ↑

#Way1:

Operations:

5 4 7 9 ↑ 8 f( ) 10 ↑ ↑ 14 ↑ ↑ ↑

#Way2:

#Ideas:

push(x)

pop( )

Operations:

5 4 7 9 ↑ 8 f() 10 ↑ ↑ 14 ↑ ↑ ↑

#Way2:

#Idea2:

push(x)

pop()

top()

size()

```
class Queue{

    void enque(int x){





    }
    void deque(int x){





    }
    int front(){





    }
    int size(){

    }
}
```

Amortized:

Ex: 5 4 7 9 deq() deq() deq() deq()