Queue

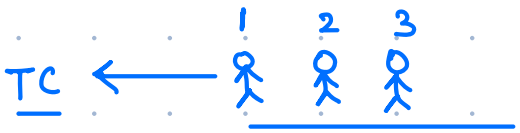
Implementation of queue using array
LL
Stack

- Perfect Number Question
- Double ended queue
- Sliding window maximum

Queue

linear data structure where element gets added from one end, but get removed from other end.

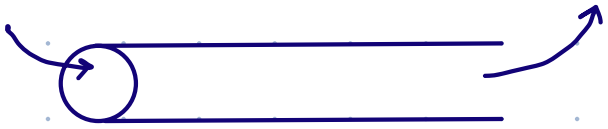
F I F O



Some real world examples —

1. Ticket counter
2. Toll plaza
3. Printer — Queue to print
4. Customer service
   → First come first serve basis

Functions :

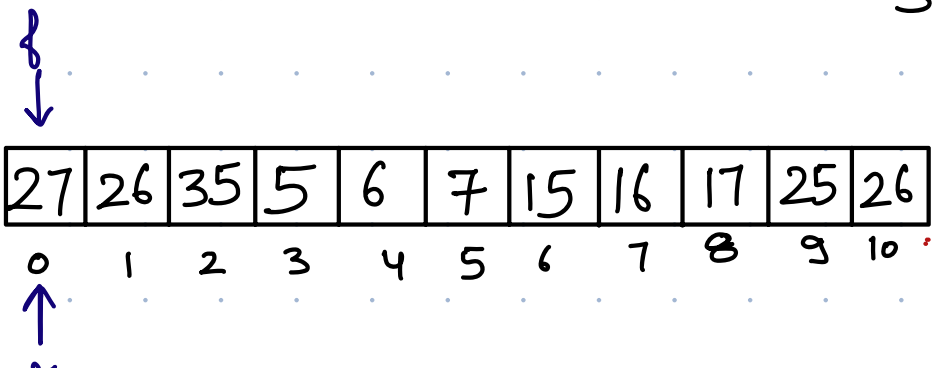


add - Enqueue()
remove - Dequeue()
top - peek()
size - size()

LL
Array
Stacks

Arrays

size = 11



f = front = -1 = index of element which just got removed

r = rear = -1 = index of element which has just been added

enqueue (19)
enqueue (26)
enqueue (35)

dequeue ()
enqueue (5)
enqueue (6)
enqueue (7)

enqueue (15)
enqueue (16)
enqueue (17)
enqueue (25)
enqueue (26)
enqueue (27) ←

f = f + 1

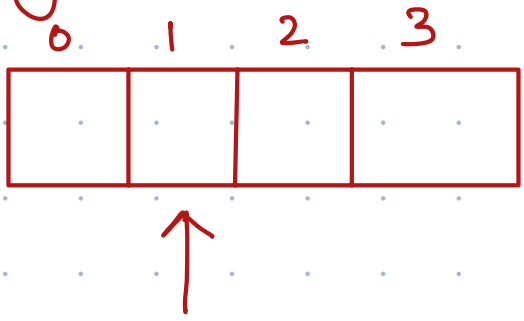
r = (r + 1) % N
A[r] = x

N = 11

pop

f + 1

How will you move?



N = 4

r = 0

1 % 4 = 1

2 % 4 = 2

3 % 4 = 3

4 % 4 = 0

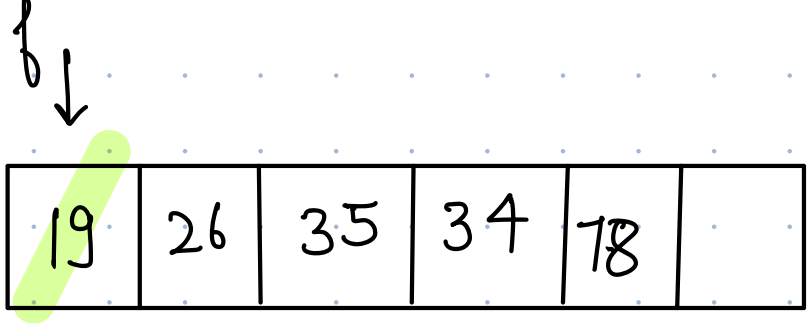
(0+1) % 4 = 1

Circular Way

f == r

Queue being full

A small mistake



empty

dequeue ()
dequeue ()
dequeue ()
dequeue ()

N

SZ

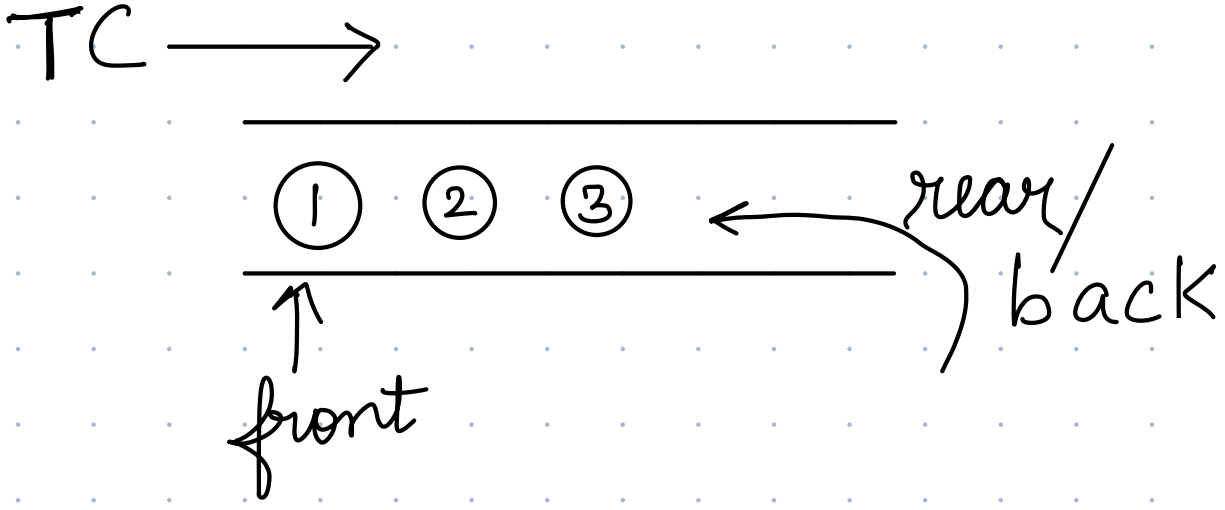
```
void enqueue (int x) {
    if (sz == N) {return;}

    r = (r+1) % N
    A[r] = x
    sz++
}
```

```
int dequeue () {
    if (sz == 0) {
        return -1;
    }
    f = (f+1) % N
    int val = A[f]
    sz--
    return val;
}
```

```
int peek() {
    if (sz == 0) { return -1; }
    int idx = (f+1) % N
    return A[idx];
}
```

```
int size() {
    return sz;
}
```



Linked List

O(1)

Insert (addLast)

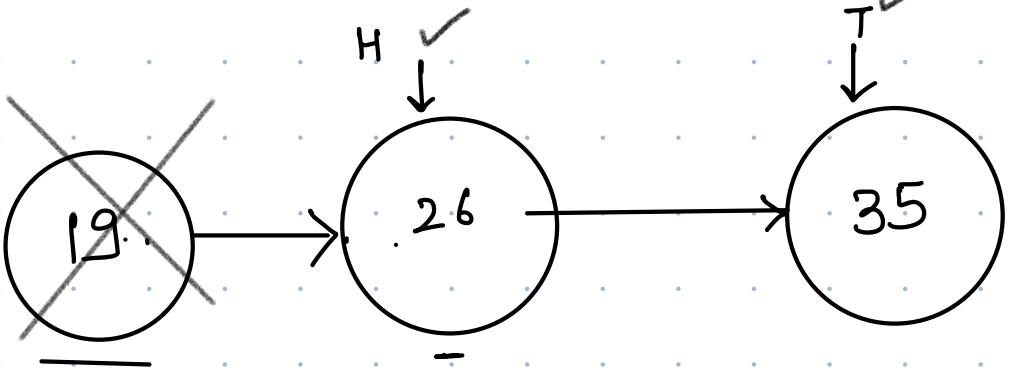
T.Next = NE

T = T.Next

SZ++

Head = ~~NULL~~ (19)

Tail = ~~NULL~~ (19)



Removal (RemoveFirst)

O(1)

H = H.Next

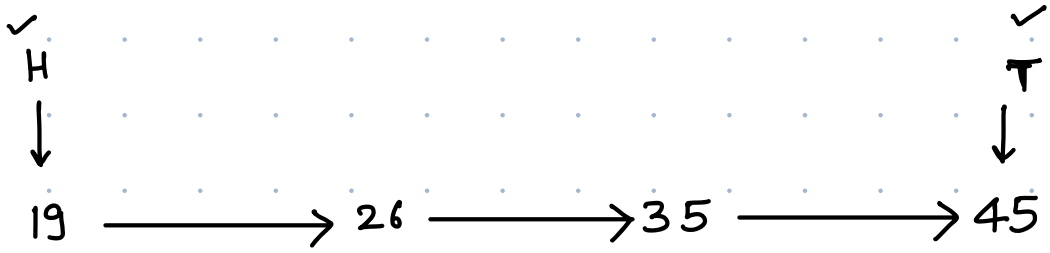
SZ--

Peek()

↳ Head

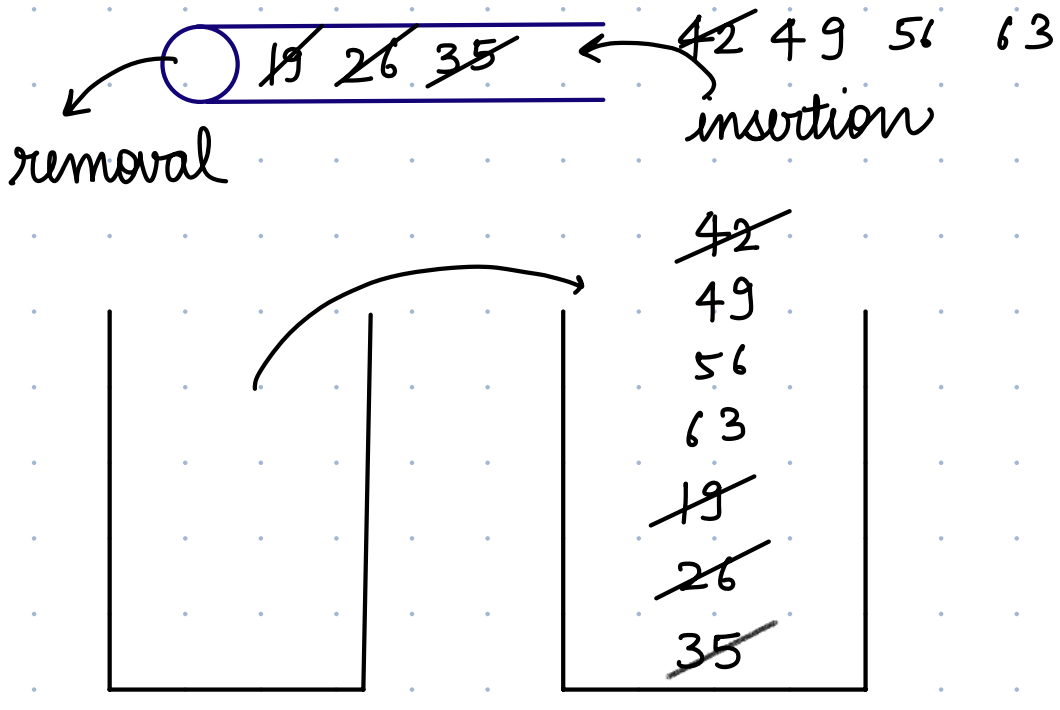
Size()

SZ



Singly L L

Queues



19
26
35
42
49
56
63

(Removal Order of queue)

enqueue( )

push into enqueue stack

dequeue( )

dequeue stack

sz == 0

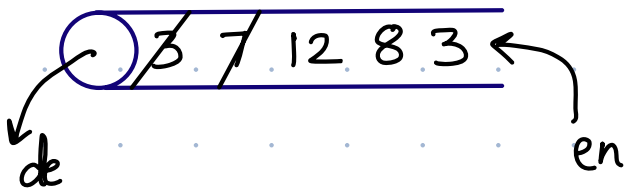
1. empty all enqueue stack and put it into dequeue stack
2. pop

sz ≠ 0

pop

Quiz:

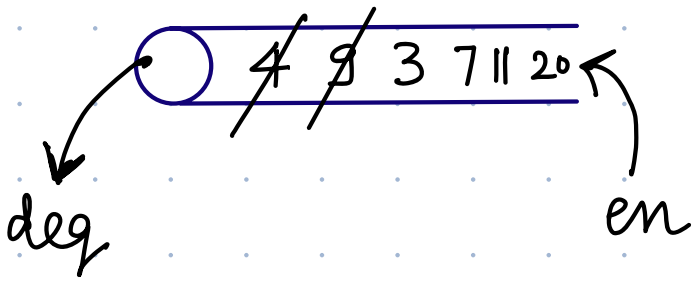
en(3)
en(7)
en(12)
deq( )
deq( )
en(8)
en(3)



12 8 3 ans

Quiz:

enq(4)
deq
enq(9)
enq(3)
enq(7)
enq(11)
enq(20)
deq()



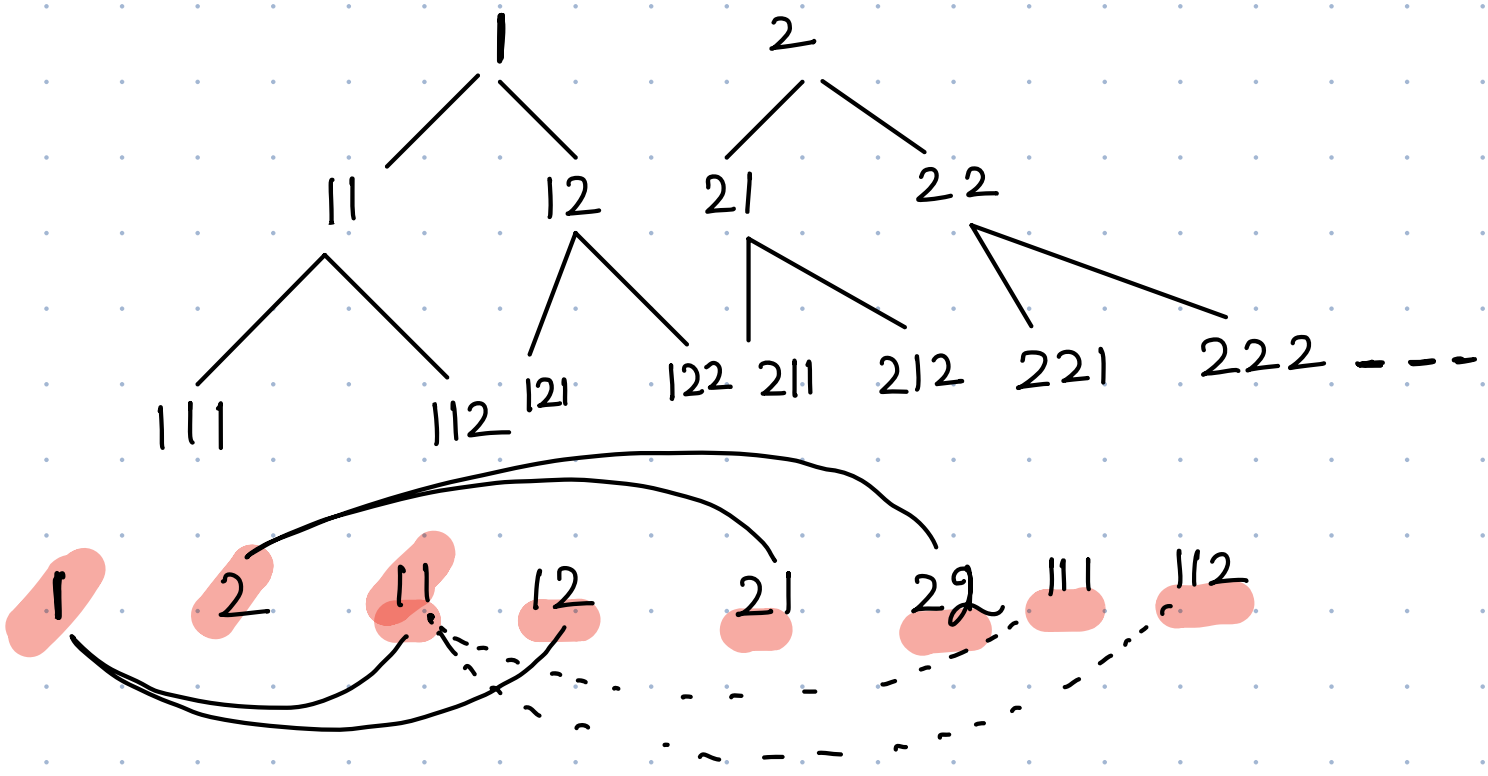
3 7 11 20

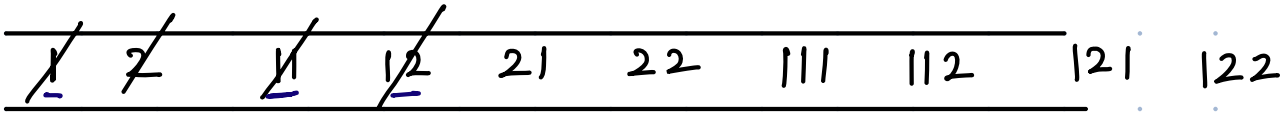
Q Perfect Number

1 2 11 12 21 22 111 112 121 122 211 212 ...

1 2 3 4 5 6 7 8 9 10 ... 221

N ⟶ Nth number of series





12 × 10 + 1 = 121

12 × 10 + 2 = 122

```
int solve ( int N) {
    if (N == 1)  return 1;
    if (N == 2)  return 2;
    Queue<int> q;
    q.enqueue(1)
    q.enqueue(2)
    i=3
    while ( i <= N) {
        rem = q.dequeue()
        a = rem x 10 + 1
        b = rem x 10 + 2
        if (i == N) {return a}
        if (i+1 == N) {return b}

        q.add(a);
        q.add(b);
        i = i + 2;

    }
}
```

TC: O(N)

SC: O(N)

Dry Run:

N = 6

1	2	11	12

i = 3 5

a = 11 21

b = 12 22 ans

Deque : Doubly Ended Queue

linear data structure
allows removal & insertion at both ends

6 important functions :

1)	Insert at front	push_front ( )
2)	back	push_back ( )
3)	delete at front	pop_front ( )
4)	delete at back	pop_back ( )
5)	front	front ( )
6)	last	back ( )

Sliding Window Maximum

Q given an integer array A and a window K.

Find the maximum in all windows

$$A = \{10 \quad 1 \quad 9 \quad 3 \quad 7 \quad 6 \quad 5 \quad 11 \quad 8\}, \quad K = 4$$

Ans = 10 9 9 7 11 11

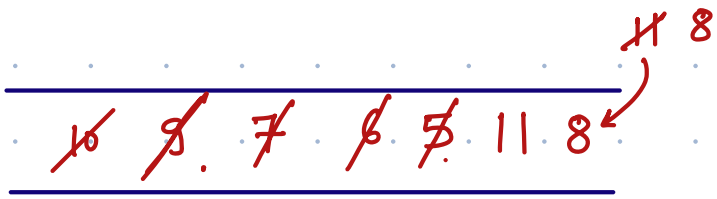
$$\{1 \quad 4 \quad 3 \quad 2 \quad 5\} \qquad K = 3$$

Ans 4 4 5

SLIDING WINDOW

K = 4

10 1 9 3 7 6 5 11 8

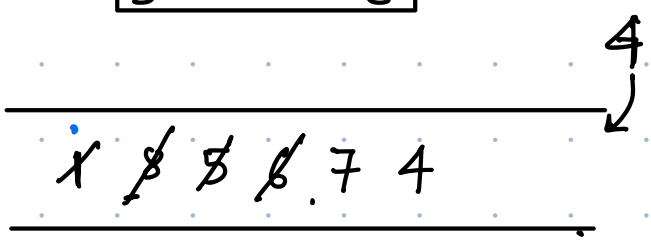


Ans = 10 9 9 7 11 11

K = 4

1 8 5 6 7 4 2 0 3

s e



ans 8 8 7 7 7 4

1) Solve first window

FIRST WINDOW = TOP of DEQUE

2) Solve each window

A[s-1] = top of Deque

remove

A[e] ↴

Deque

ans = TOP of DEQUE

```
void maxwindow (int[] A, int k) {

    Deque<int> dq

    // FIRST WINDOW
    for( int i=0; i<K; i++) {
        while(dq.size()>0 &&
              A[i] > dq.back()){
            dq.pop_back()
        }
        dq.push_back(A[i])
    }

    print (dq.front())

    // Next windows
    S = 1, e = K
    N = A.length
    while ( e < N) {

        // Handle S-1
        if ( A[S-1] == dq.front()){

            dq.pop_front();
```

```
}
// Handle e
while( dq.size()>0
        && A[e]>dq.back()){
    dq.pop_back();
}
dq.push_back(A[e]);
// give the ans
print(dq.front());
```

s++
e++
} move to next window

}

}

Every element —— touched twice

going in } deque
going out }

TC: O(N)

SC: O(K)