Agenda

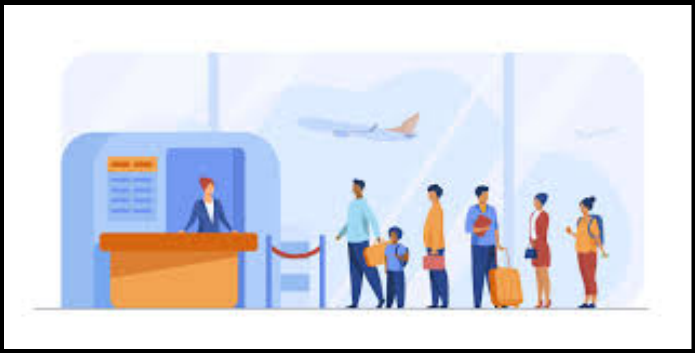
Queues Basics

Implementation

Nth no. using only 1,2, or 3 as digits

Find Nth perfect no.

remove/front



insert/rear

Queue → FIFO

Real Life

1) Playlist
2) Movie Ticket
3) Customer Care
4) Bus Ticket line

Operations →

O(1)

1) Enqueue (x) : Inserts the ele into the queue at rear end.

2) Dequeue() : Remove the element out of the queue from front end

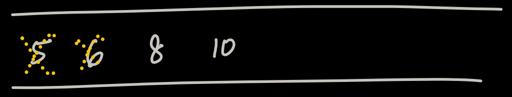
3) front() : returns the ele present at front of queue

4) back() : returns the ele present at end of queue

5) isEmpty() : return T/F

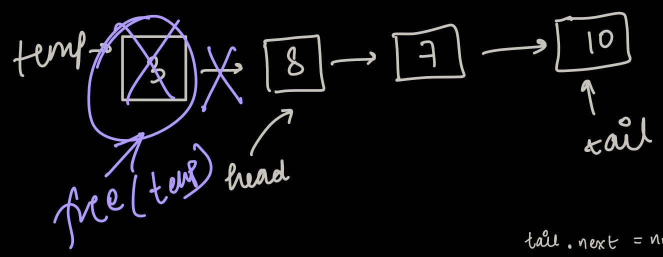
Example

```
                                             false                8      10
  ↓       ↓       ↓       ↓        ↓          ↓           ↓       ↓      ↓
E(5)    E(6)    E(8)    E(10)     D()     isEmpty()      D()     f()    b()
```



Implementation using LL

```
      ↓   ↓   ↓    ↓
  3   8   7   10   deq()
```



tail . next = new node ;
tail = tail.next ;

```
int front ( ){
    if(head == NULL){
        print(" head is NULL")
        return -1 ;
    }
    return head.data
}
```

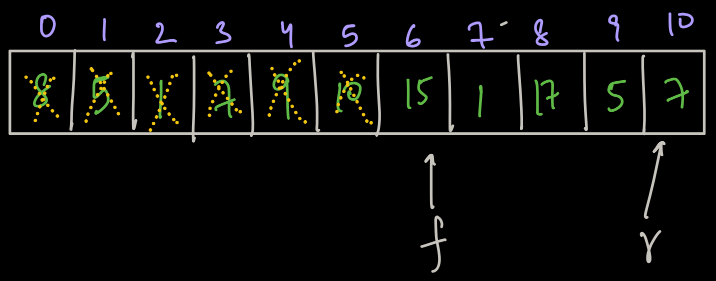
```
void enqueue ( x) {
    Node new_node = new Node(x);
    if ( head == NULL){
        head = tail = new_node ;
        return
    }
    tail.next = new_node ;
    tail = tail.next;
}
```

```
int back ( ){
    if(head == NULL){
        print(" No data present");
        return -1
    }
    return tail.data
}
```

```
int dequeue ( ){
    if (head == NULL) { return -1; }
    if(head == tail){
        temp = head
        head = tail = NULL
        free(temp) ;
    }
    temp = head
    head = head.next
    temp.next = NULL
    free(temp)
}
```

Queues using Arrays

front (f)
rear (r)



```
enqueue (x) {
    if (r == N-1) {
        print("overflow")
        return
    }
    if(f == -1){
        f=0
    }
    r++ ;
    A[r] = x ;
}
```

```
dequeue () {
    if (f == -1) {
        print ("underflow")
        return
    }
    else if(f == r ){
        f = r = -1
    }
    else f++
}
```

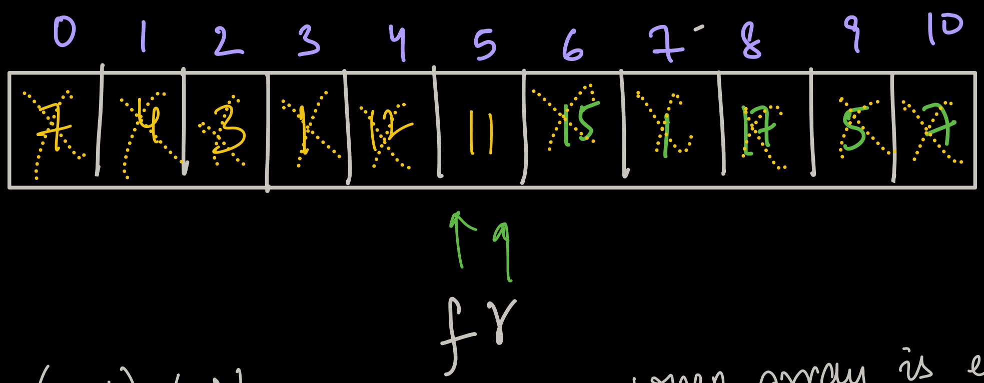
f → points to the first ele of queue

r → points to the last ele of the queue

```
front( ) {
    if(f == -1) return -1
    return A[f];
}
```

```
back ( ) {
    if (f == -1) { return -1 }
    return A [r] ;
}
```

Circular Queue



$(r+1)\%N$

$(f+1)\%N$

when array is empty

$f = -1$ & $r = -1$

when the array is full

$(r+1)\%N == f$

when only single ele

if $f != -1$

and $f == r$

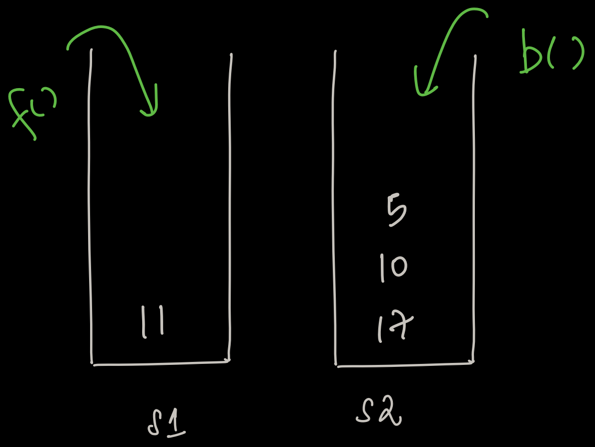
$f = -1$

$r = -1$

Implementation using Stacks

↳ push, pop, top

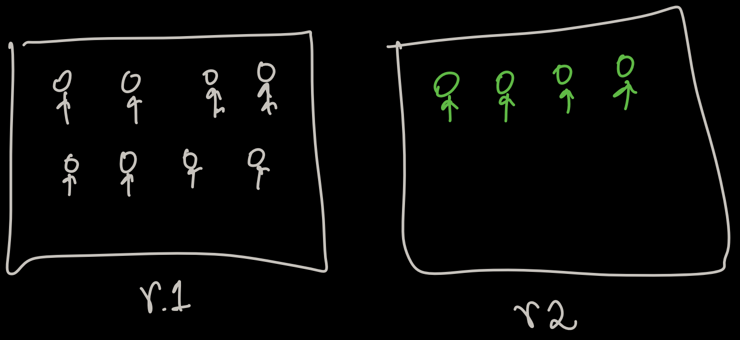
7 8 4 2 -1 d() d() 3 10 d() 17 d() d() d() 11



~~7~~ ~~8~~ ~~4~~ ~~2~~ ~~-1~~ 5 10 11

enqueue() → O(1)

dequeue() → O(N)



Doctor

enqueue → O(1)

dequeue → O(1) amortized

for 1 dequeue if it is N iteration.

Then for next N dequeues, it's 1 item.

Total → N iterations

```
s1 , s2

void equeue(x) {
    S1 . push(x)
}
```

```
void dequeue( ) {
    if(s2.size() == 0){
        while(s1.size()>0){
            s2.push(s1.pop())
        }
    }
    if(s2.size()>0){
        s2.pop()
    }
}
```

cat who is in the air ?

billionaire

billi - on - aire

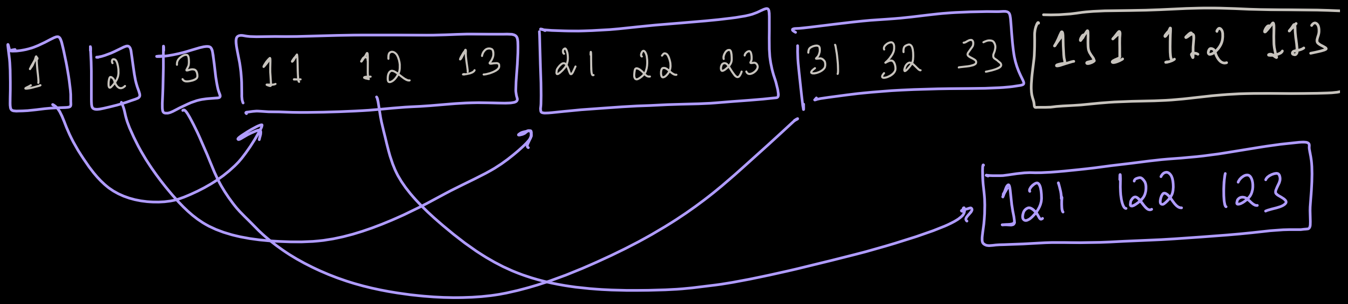
Que

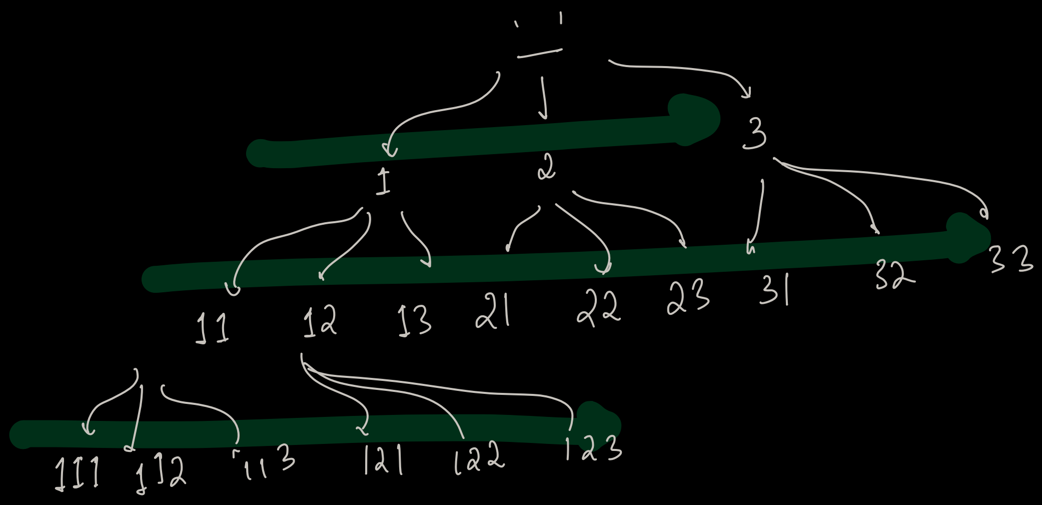
Kth no. that can be formed with digits

1, 2, 3

K = 5th ⟶ 12
K = 7th ⟶ 21

1
2
3
11
12
13
21
22
23
31
32
33
111
112





1	2	3	4	5	6	7	8	9	10		
~~1~~	~~2~~	~~3~~	~~11~~	~~12~~	~~13~~	~~21~~	~~22~~	~~23~~	31	32	33

```
queue<string> q;
q.insert("1");
q.insert("2");
q.insert("3");
count = 3;
deleted = 0

while(count < K){
    string s = q.front()
    q.pop()
    delete ++;
    q.insert(s+'1');
    q.insert(s+'2');
    q.insert(s+'3');
    count += 3
}
```

K=10

cnt

~~3~~

~~6~~

~~9~~

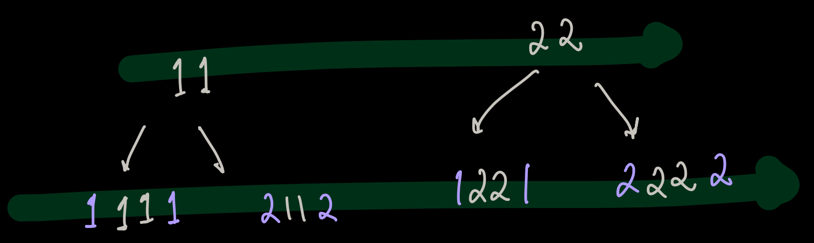
12

```
//Remove starting ele to get the kth ele.
while(deleted < K-1){
    q.pop()
    deleted++
}
return q.front();
```

Que. Nth perfect Number

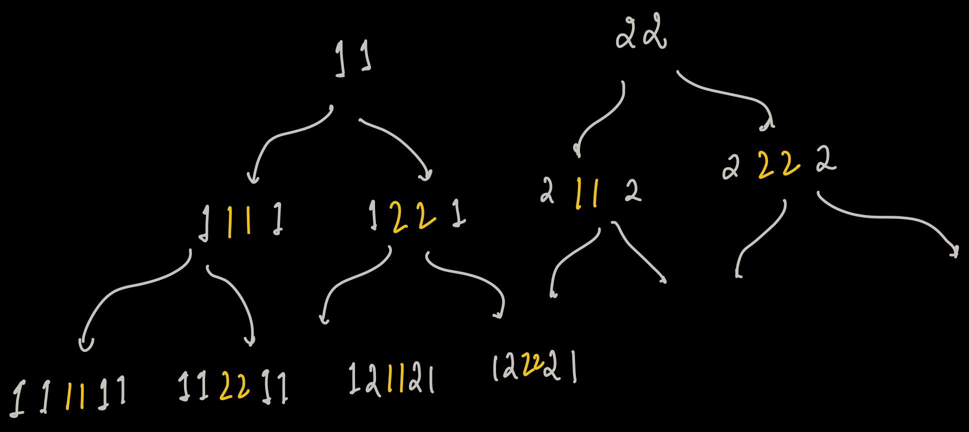
- even length
- palindrome
- digit (1, 2)

11 22 1111 1221 2112 2222 111111 . . . .



Are we getting ele in sorted order ?

No ✗



~~11~~ ~~22~~ 1111 1221 2112 2222 . . .

way 2

11

22

1111

1221

2112

2222

111111

112211

~~1~~ ~~2~~ ~~11~~ 12 21 22 111 112

Doubt

1	2	3	4	5	6	7	8		
~~1~~	~~2~~	~~11~~	~~12~~	21	22	111	112	121	222

$112 \xrightarrow{rev} 211$

112211