Today's agenda

- Queue basics
- Reverse first K ele in queue
- implement queue using stack
- Kth number using only 1 & 2

// Introduction

LIFO

↳ First In First Out (FIFO)

Real life ex:

(i) Line

(ii) task Scheduling

Syntax:

↳ Queue <Integer> q = new LinkedList <>();

name

Operations:

(i) q.add (or) → insert or at the end of queue.

(ii) q.remove() → delete element from front.

(iii) q.Peek() → return the front element.

(iv) q.size() → No. of elements in queue.

Q) Reverse first K elements

↳ Given a Queue, Reverse it's first K elements.

Eg: K=4 | 3 10 2 12 | 19 6 8 10 14

Idea

↳ Put the first K elements in the Stack.

Steps :

↳ Push K elements Queue → Stack.

Step 2:

↳ Put K elements back from Stack → Queue.

Step 3:

↳ Remove the first n-k elements from the front and add at the end.

//Psuedo code

```
Queue<> ReverseKelements (Queue <>q, int k ){
    Stack <Integer> s = new Stack<>();
    int n= q.size();
    for (int i=0; i<k; i++){
        s.push(q.remove());
    }

    for (int i=0; i<k; i++){
        q.add(s.pop());
    }

    for (int i=0; i<n-k; i++){
        q.add(q.remove());
    }

    return q;
}
```

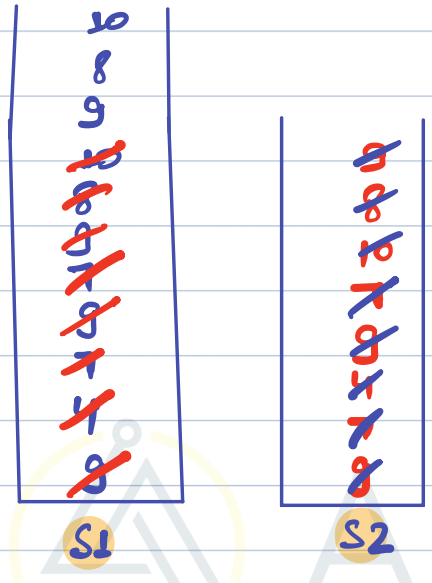
T.C: O(n)

S.C: O(k)
↓
k=n

Q) Implement Queue (FIFO) using Stacks (LIFO)

op: 5 4 7 9 rem sum 8 10 rem rem 14 rem

rem → 5 | sum → 4 | rem → 7



Idea1 → add efficient

add(x): add x in S1.
↳ O(1)

remove(): → O(N)

(i) move n-1 elements S1 → S2

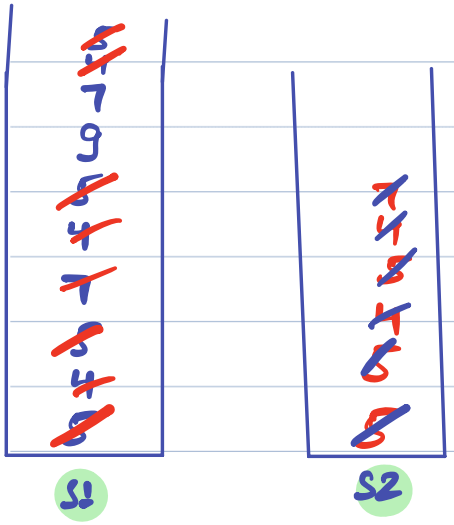
(ii) The remaining element of S1 is your ans.

(iii) move all elements from S2 → S1

Idea2 → remove efficient

op: 5 4 7 9 rem sum 8 10 rem rem 14 rem

rem → 5 | sum → 4



add(x): → O(N)

(i) move all elements S1 → S2

(ii) add x to S1.

(iii) Put back all elements S2 → S1

remove(): → O(1)
↳ remove the top of S1.

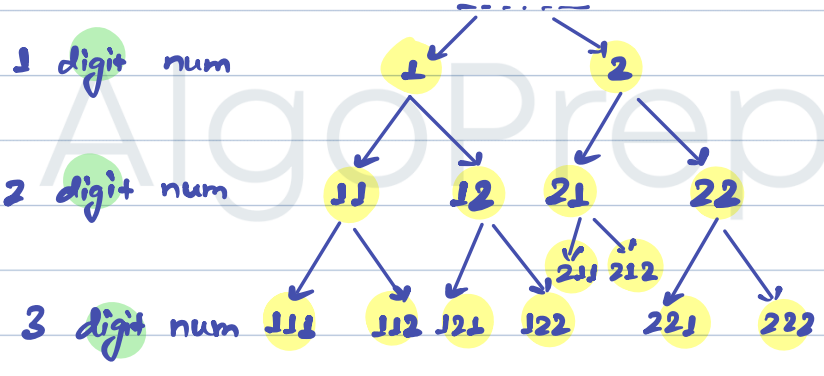
Q) Kth Number

↳ Generate Kth number in series using digits 1 and 2 only.

K=5 1 2 11 12 21

K=7 1 2 11 12 21 22 111

//Idea



//Algorithm

K=5

1 2 11 12 21 22 111 112 121 122

↳ ans = 21

//Pseudo code

```
String KthNumber(int k){
    Queue<String> q;
    q.add("1");
    q.add("2");

    String ans:"";
    for(int i:1; i<=k; i++){
        String temp = q.remove();
        if(i==k){ ans = temp; }
        q.add(temp+"1");
        q.add(temp+"2");
    }

    return ans;
}
```

digit length

T.C: O(K * n)

S.C: O(K)

Break till 9:40 pm

Q)

↳ Generate Kth Palindrome number in series using digits 1 and 2 only

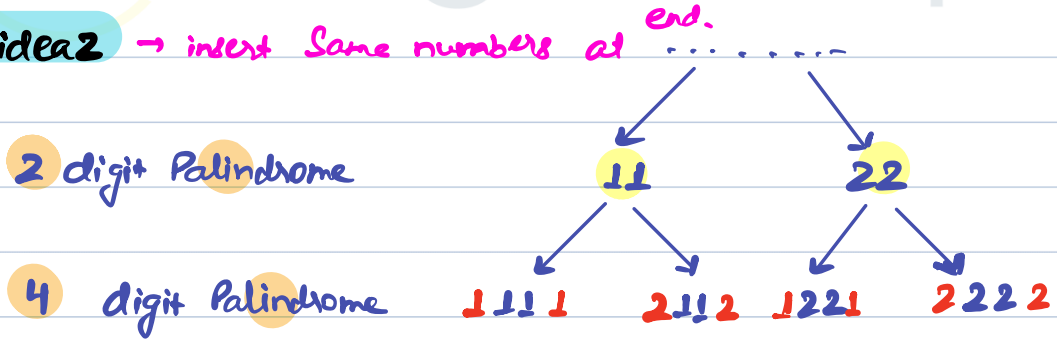
Note: only consider even digit numbers.

K:5 : 11 22 1111 1221 2112

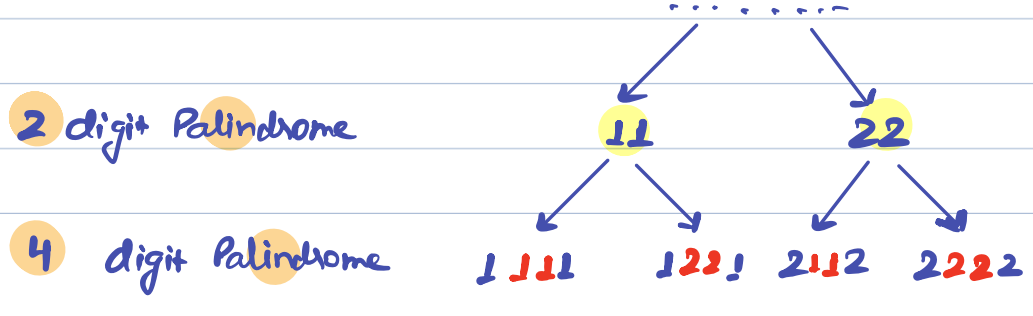
//idea 1

↳ keep generating numbers using 1 and 2, check for even digit Palindrome and return the Kth one.

//idea 2 → insert same numbers at end. . . . . .



→ insert some numbers in middle. . . . . .



//Pseudo code

```
String kthNumber(int k){
    Queue<String> q;

    q.add("11");
    q.add("22");
    String ans:"";
    for(int i=1; i<=k; i++){
        String temp= q.remove();
        if(i==k){ans= temp;}
        String left = temp.subString(0, temp.length()/2);

        String right: temp.subString(temp.length()/2, temp.length());

        q.add(left + "11" + right);
        q.add(left + "22" + right);
    }

    return ans;
}
```

digit length

T.C: O(k * n)

S.C: O(k)

inc. exc.