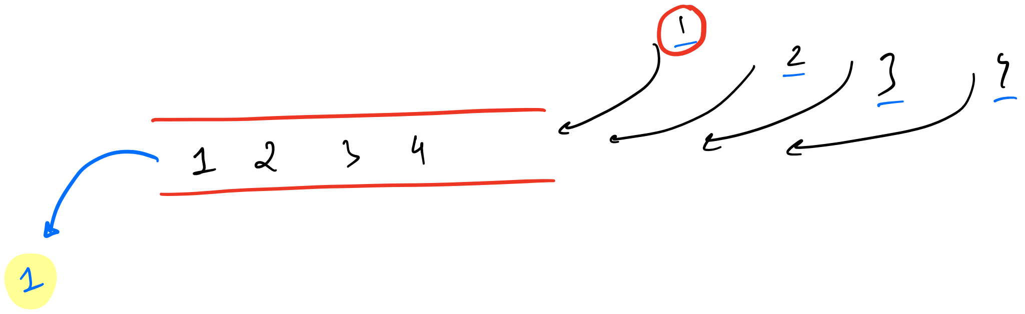
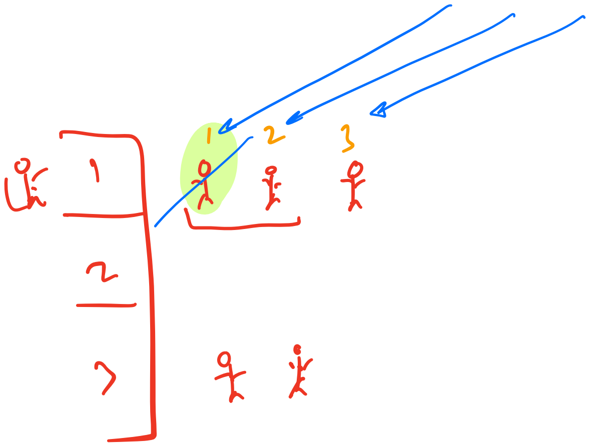
Queue

FIFO

first In First Out





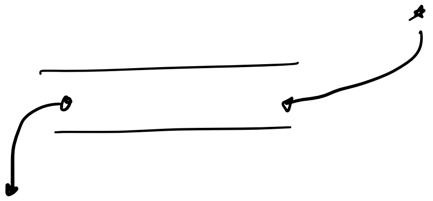
Ops of Queue

1) enqueue (x) : Insert x at the back of the Queue

2) dequeue () : Removing the front most.

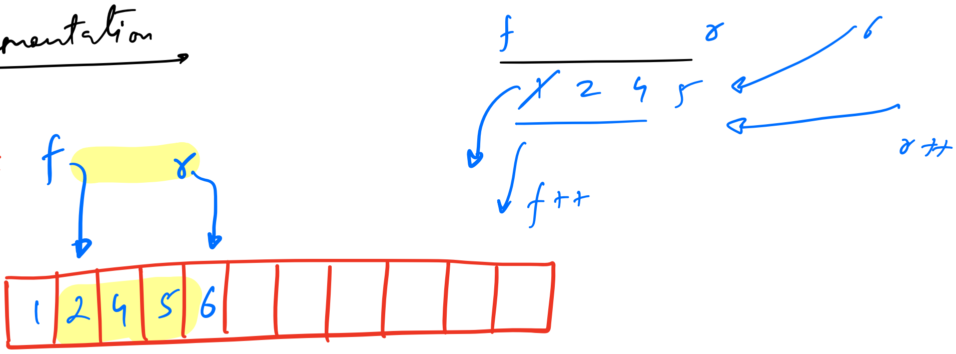
3) front () : ret the frontmost.

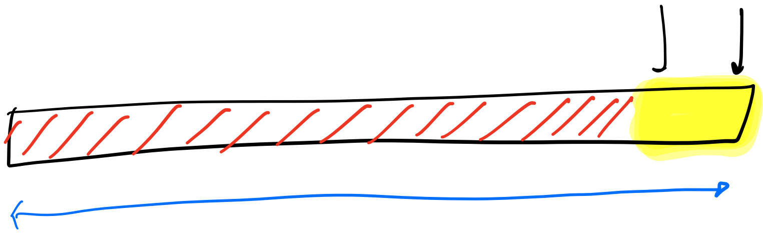
4) size () : ret the # of elements ..



① Implementation

⊙ Array:



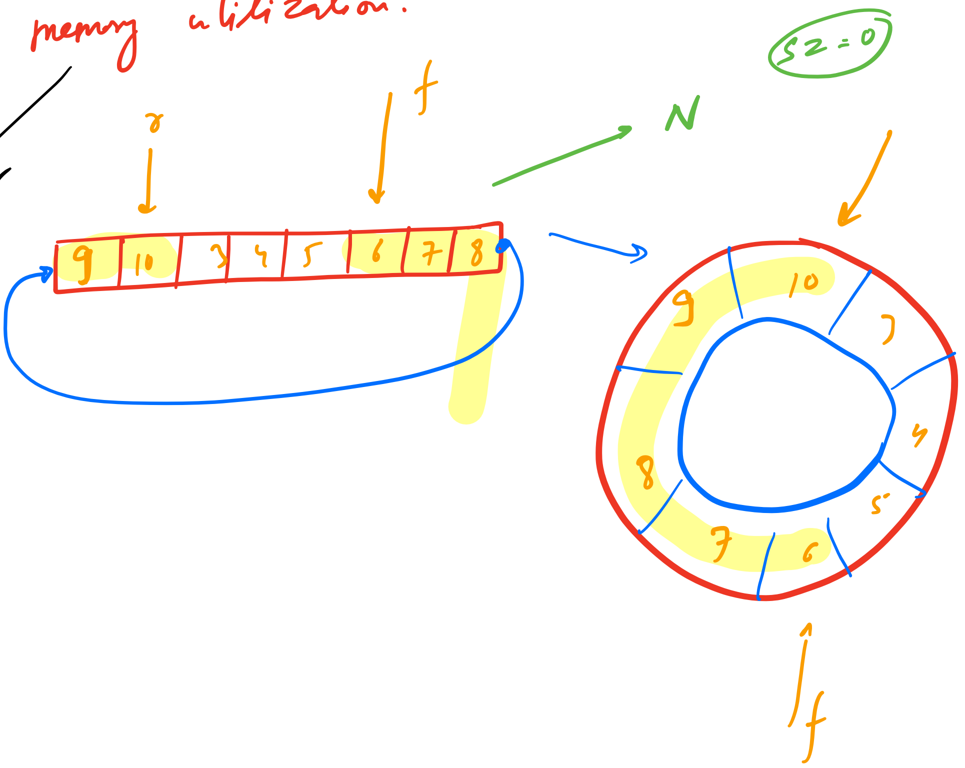


Problem:

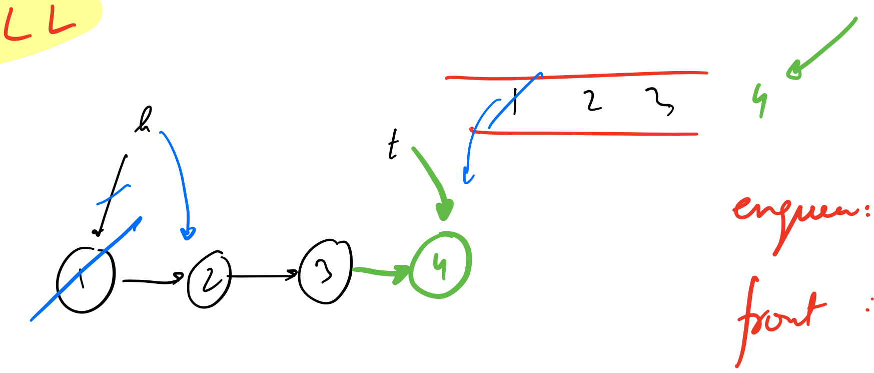
1) Need to know the # of enqueue() ! → Dynamic ✓ Array!

2) Proper memory utilization.

CIRCULAR ARRAY ✓



⊙ LL



$sz = \cancel{0}\ \cancel{1}\ \cancel{2}\ \cancel{3}\ \cancel{2}\ 3$

enqueue : O(1)

front : O(1)

dequeue : O(1)

size() : O(1)

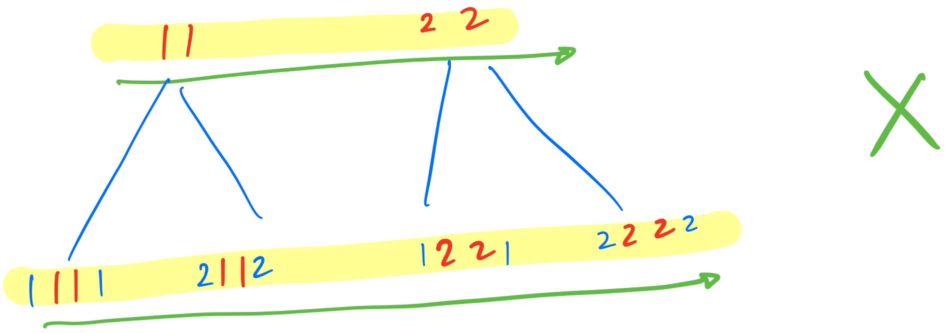
Q Given N. Generate first N perfect No's.

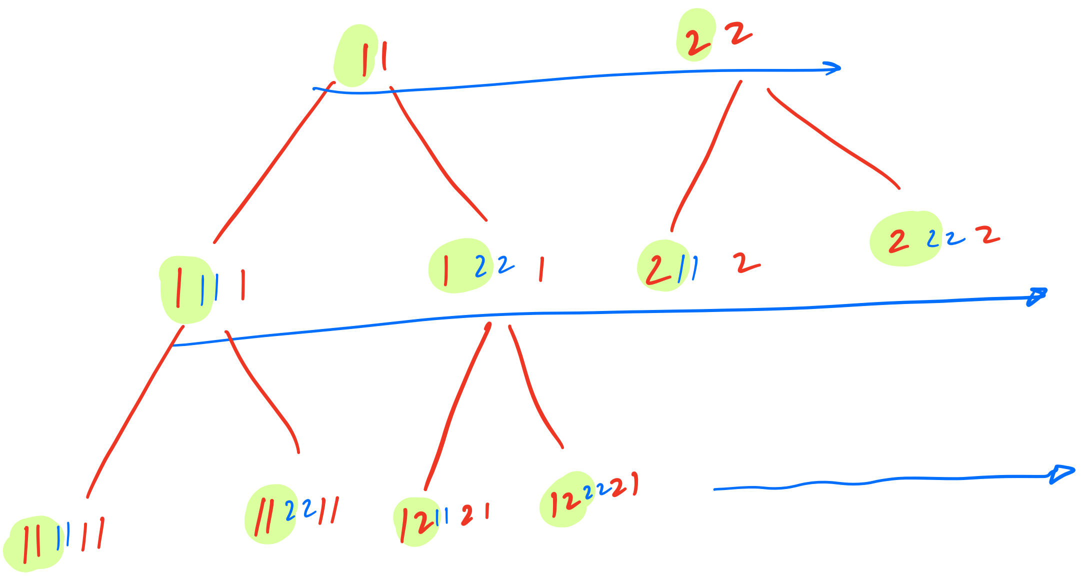
→ Even no. of digits

→ palindrome

→ digits ∈ (1, 2)

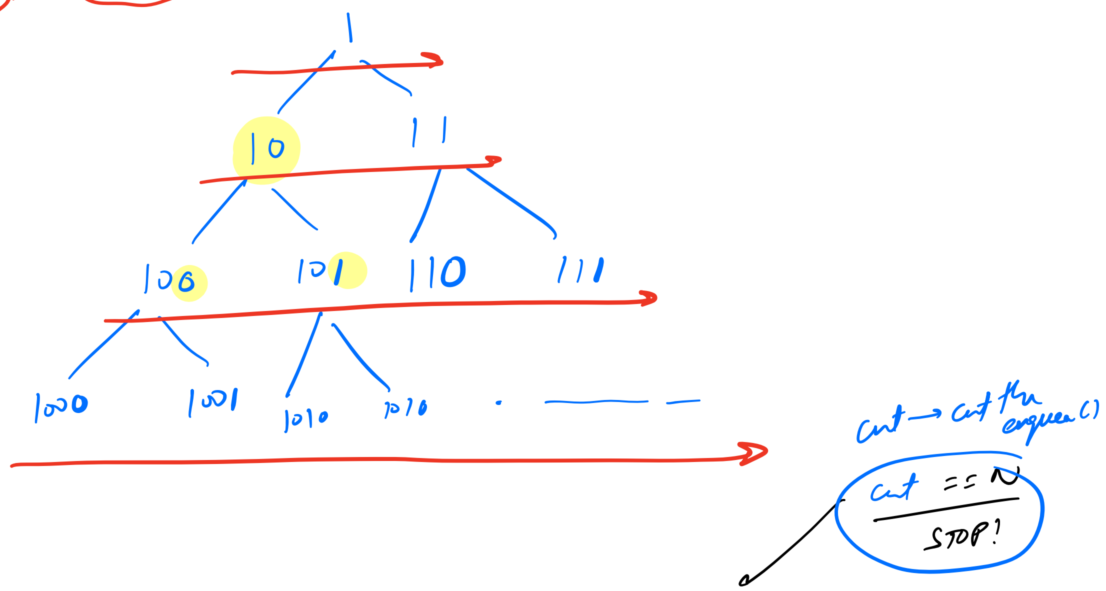
11, 22, 1111, 1221, 2112, 2222, 111111

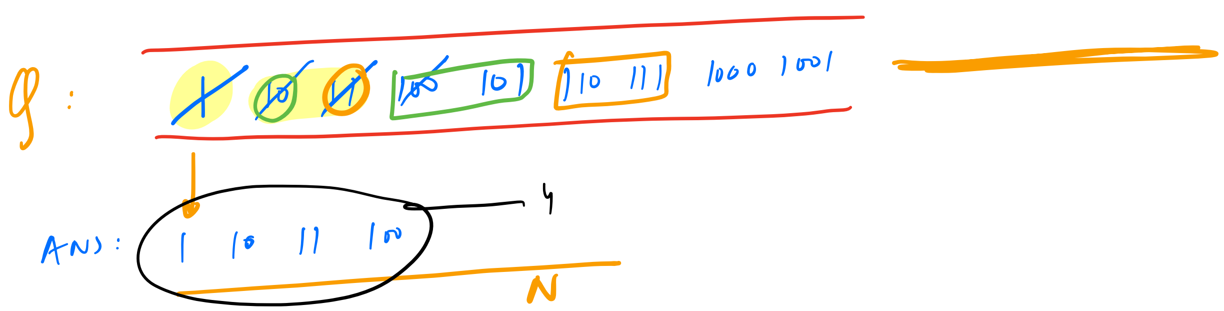


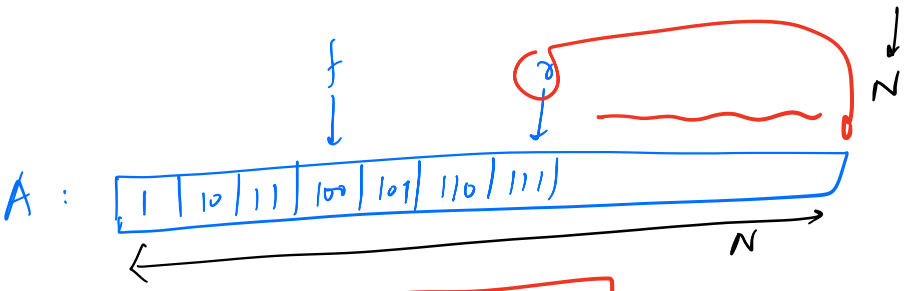


Q Generate all binary no's from 1 — N

1, 10, 11, 100, 101, 110, 111, 1000, 1001, 1010 —







TC : O(N)

SC : O(1)

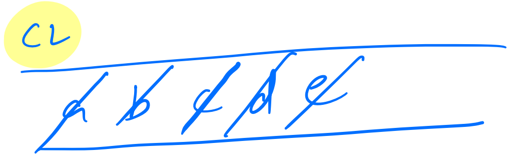
Q Given a stream of characters, after adding every character to a string, find the first non-repeating char so far.

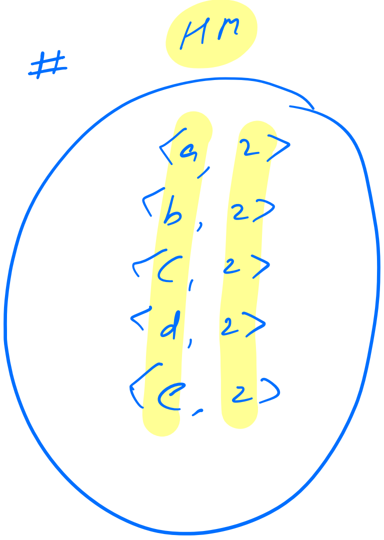
stream: a b c a d e d b e c

→ a a a b b b b c c #

a b c a d e d b e c

a a a b b b b c c #





```
CL: Queue
HashMap <char, int> hm;

f (i=0; i<N; i++) {                    → N
    ch = S[i];
    hm[ch]++;
    if ( hm[ch] == 1) {
        CL.enqueue(ch);                → K : 26
    }
    while ( ! CL.isEmpty() && hm[CL.front()] > 1)
        CL.dequeue();                  → K : 26

    if(!CL.isEmpty())
        ANS(i) = CL.front();
    else  ANS(i) = '#';

}
```

$$\boxed{TC = O(N+K)}$$

K → Alphabet Set Size → 26

$$\boxed{SC = O(\min(N, K))}$$

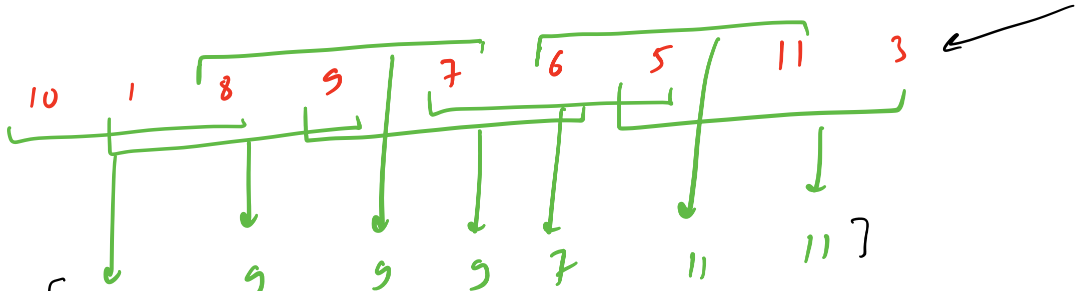
→ K / O(1)

SLIDING WINDOW MAX

Q Given an Array A.
Find the max. element for all SAs of size K.

K=3

A: 10 1 8 9 7 6 5 11 3



ANS : [10 9 9 9 7 11 11]

I) BF

∀ SA → N-K+1

find MAX → O(K)

$$\boxed{(N-K)K}$$

$$\sim N^2 \; : \; K : N/2$$

II) TreeMap

$$\boxed{TC = O(N \log K)}$$

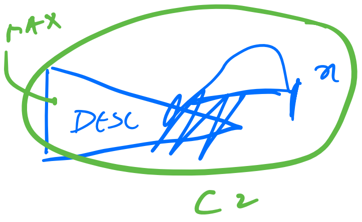
III

K=3

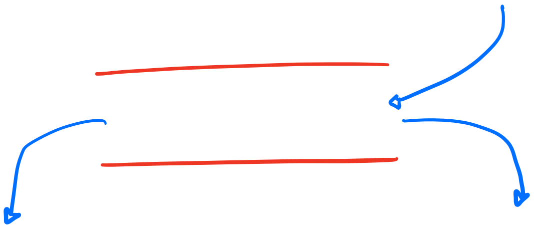
10 1 8 9 7 6 5 11 3

ANS : 10 9 9 9 7 11 11

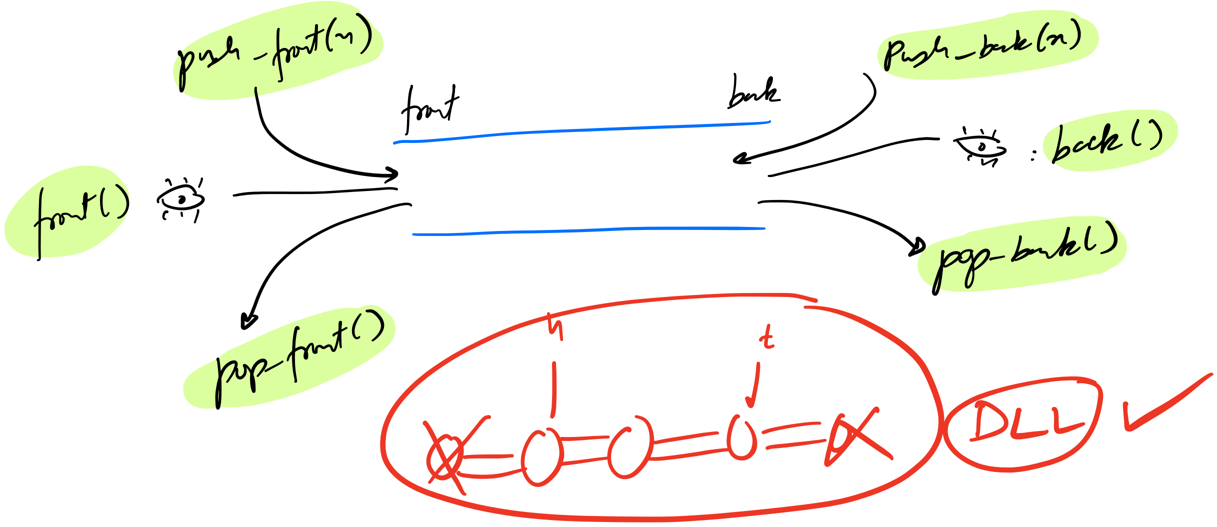
CL : ~~10~~ ~~1~~ ~~8~~ ~~9~~ ~~7~~ ~~6~~ ~~5~~ 11 3



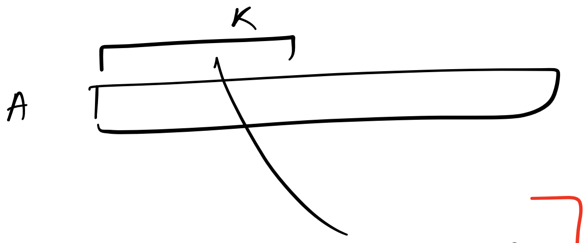
Requirement of CL ?



DEQUE : Doubly Ended Queue



```
//A[N], K
deque<int> CL;
vector<int> ans;
```



```
f(i=0; i<K; i++) {
    while(!CL.isEmpty() && A[CL.back()] <= A[i]) {
            CL.pop-back();
    }
    CL.push-back(i)
}
ans.push_back(A[CL.front()]);
```

O(K)

```
L=1, R=K;
while(R<N) {
    if(CL.front() == ~~A[~~L-1~~]~~) {
            CL.pop-front();
    }
    while(!CL.isEmpty() && A[CL.back()] <= A[R]){
            CL.pop-back();
    }
    CL.push-back(R);
    ans.push_back(A[CL.front()]);
    L++; R++;
}
```

N-K

$\boxed{TC = O(N)}$

$\boxed{SC = O(K)}$

0	1	2	3	4	5	6	7	8
10	1	7	1	5	6	5	11	3

K = 3

ANS: 10 7 7

CL: ~~0~~ ~~1~~ 2 ~~3~~ 4
(INDEX)