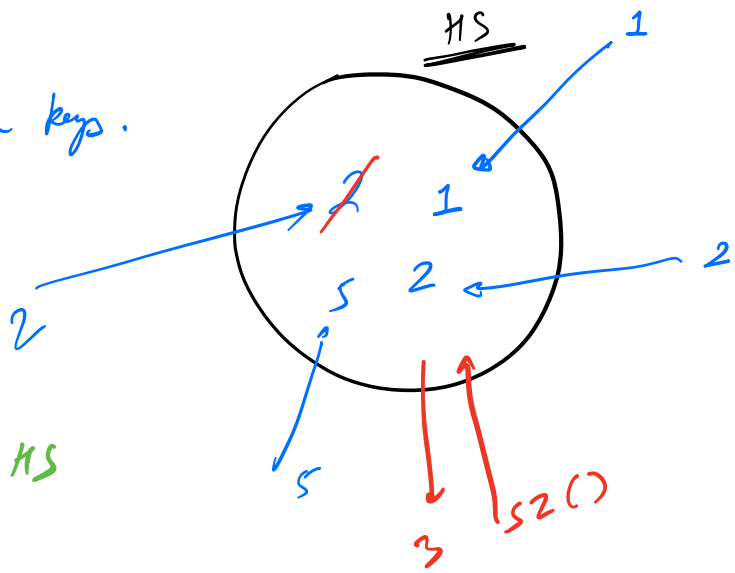
Hashing - 1

⊙ HashSet < key >

→ D.S. contains Unique keys.



f^n's

1) Insert(key)
2) check if some key is in HS
3) Remove (key)
4) size → # of keys !

~~TC = O(1)~~ ∀ ops.

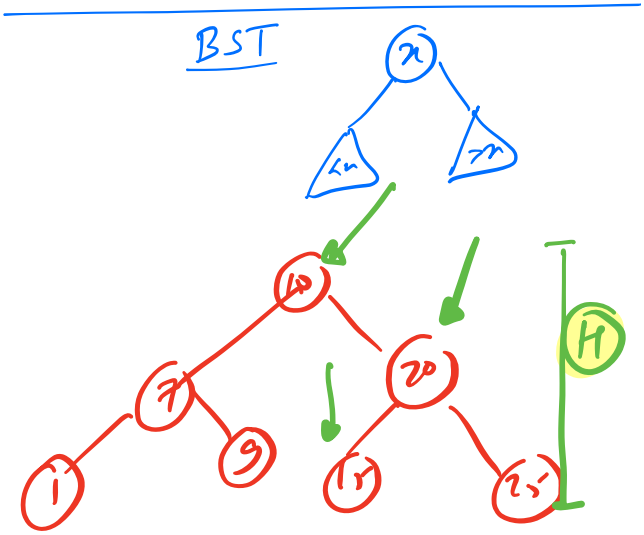
HS does NOT maintain ANY order !

Tree Set : Maintains ORDER (SORTED: ASC)

f^n's

SAME AS ABOVE

BST



TC = O(lg N) : Search, Insert, delete

BBST [Balanced BST]

- AVL
- Red Black tree

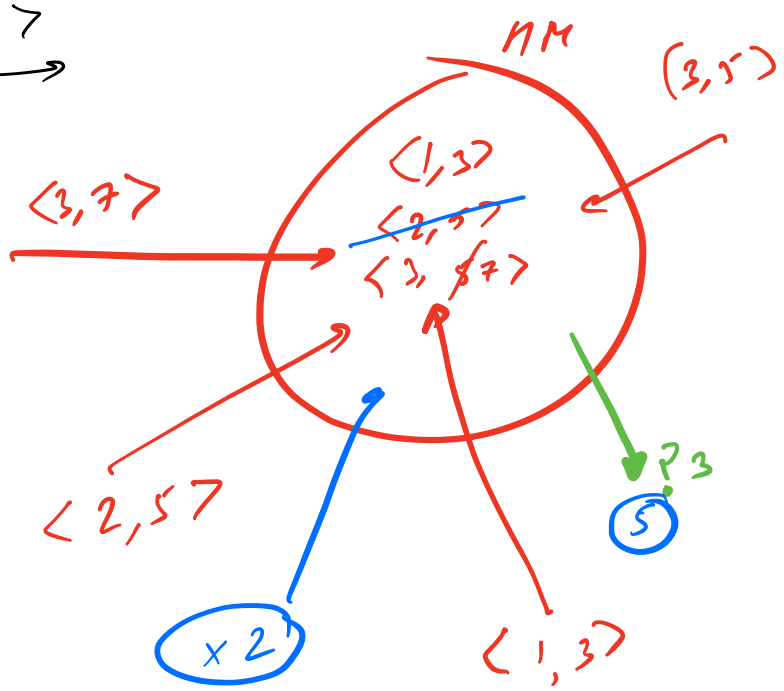
$H \rightarrow \log_2 N$

Internally uses a BBST !

HashMap <key, value>

→ DS which contains <K V> pair.

Keys are UNIQUE.



f's

→ insert(K, V)

→ get(K)

→ del(K)

→ Size()

TC = O(1) for ops

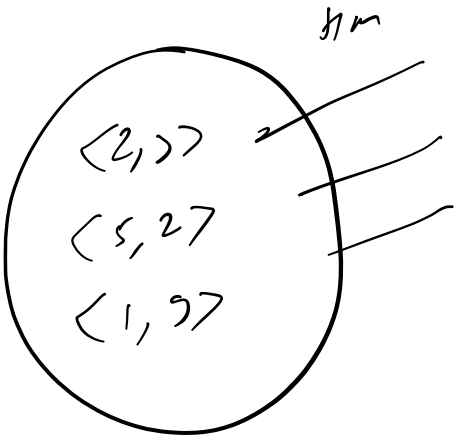
Keys are NOT ordered!

TreeMap <K, V>

Keys are ordered!

sorted: ASC

<1, 7>
<2, 3>
<5, 2>



f"s

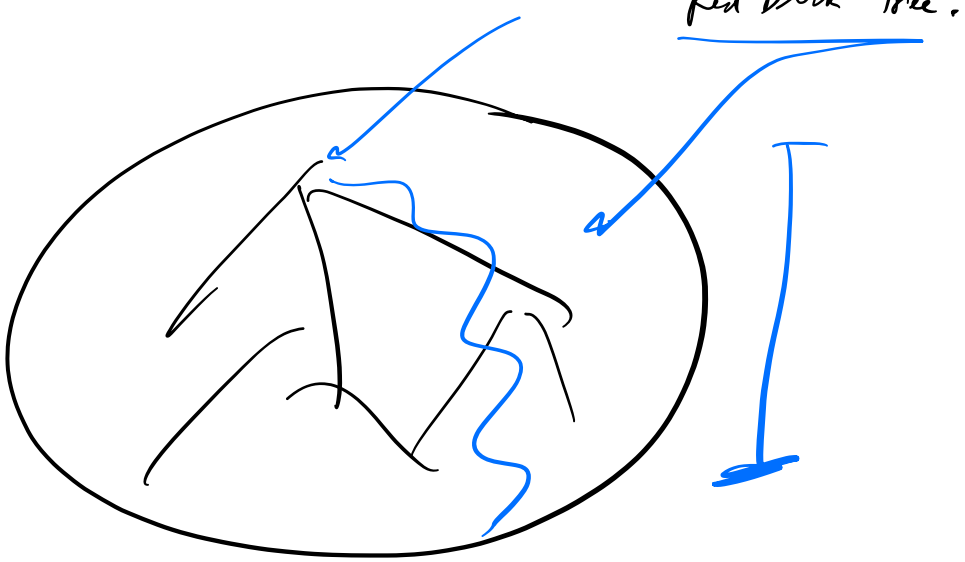
ALL SAME as ABOVE

TC = O(lg N)

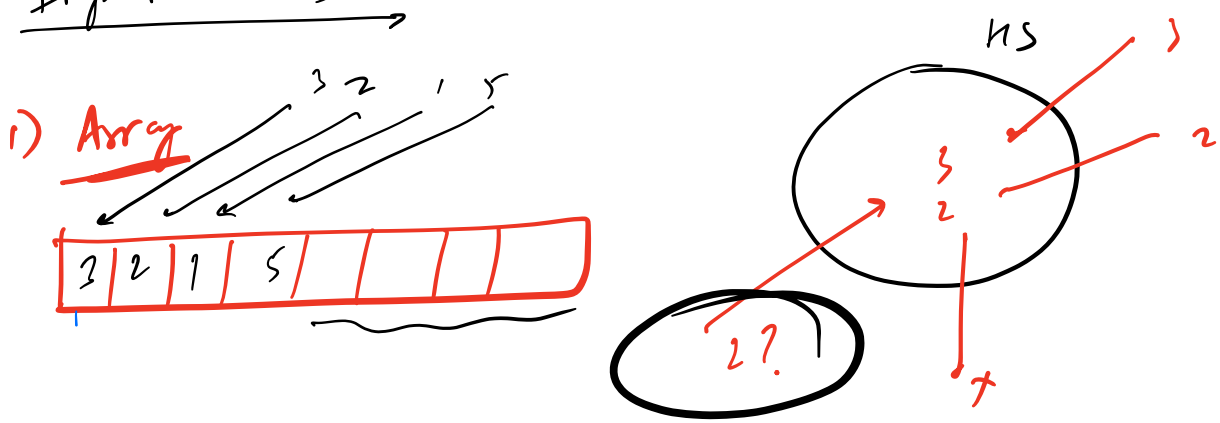
In del get

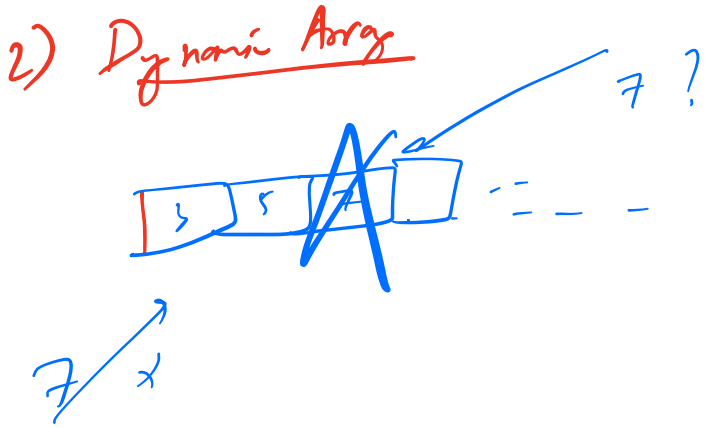
Internal implementation is BST

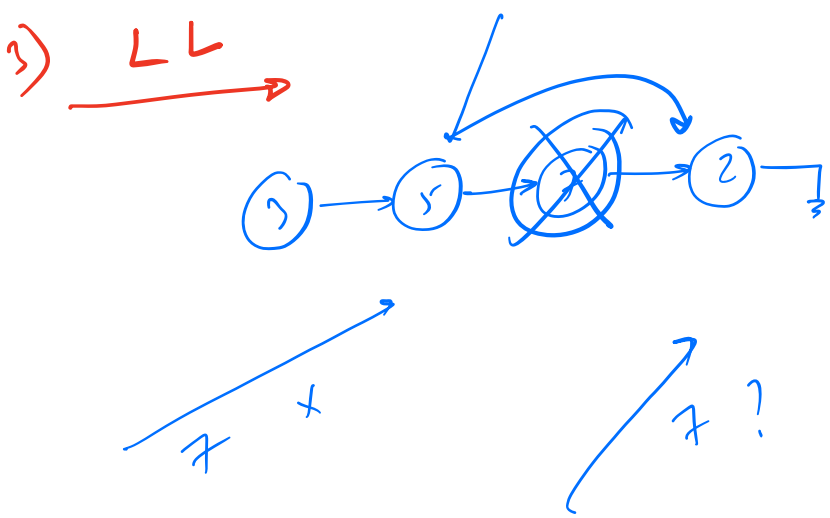
Red blck tree.



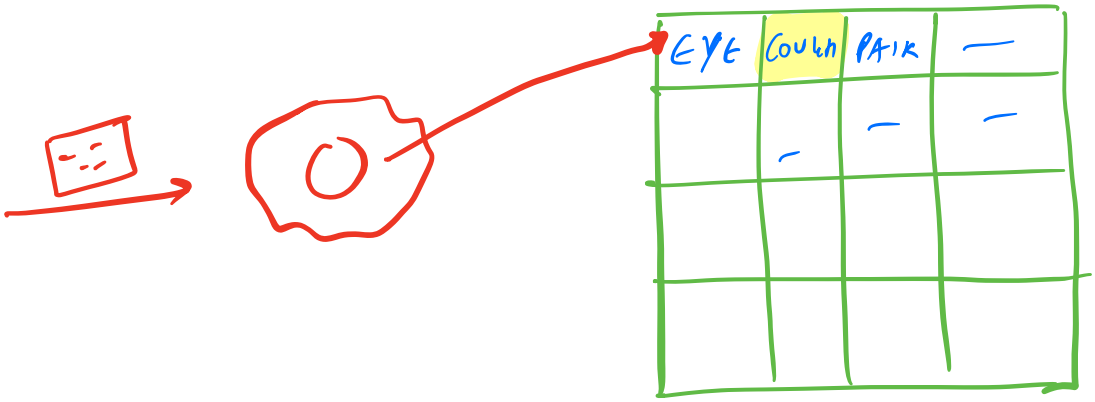
Implement MS



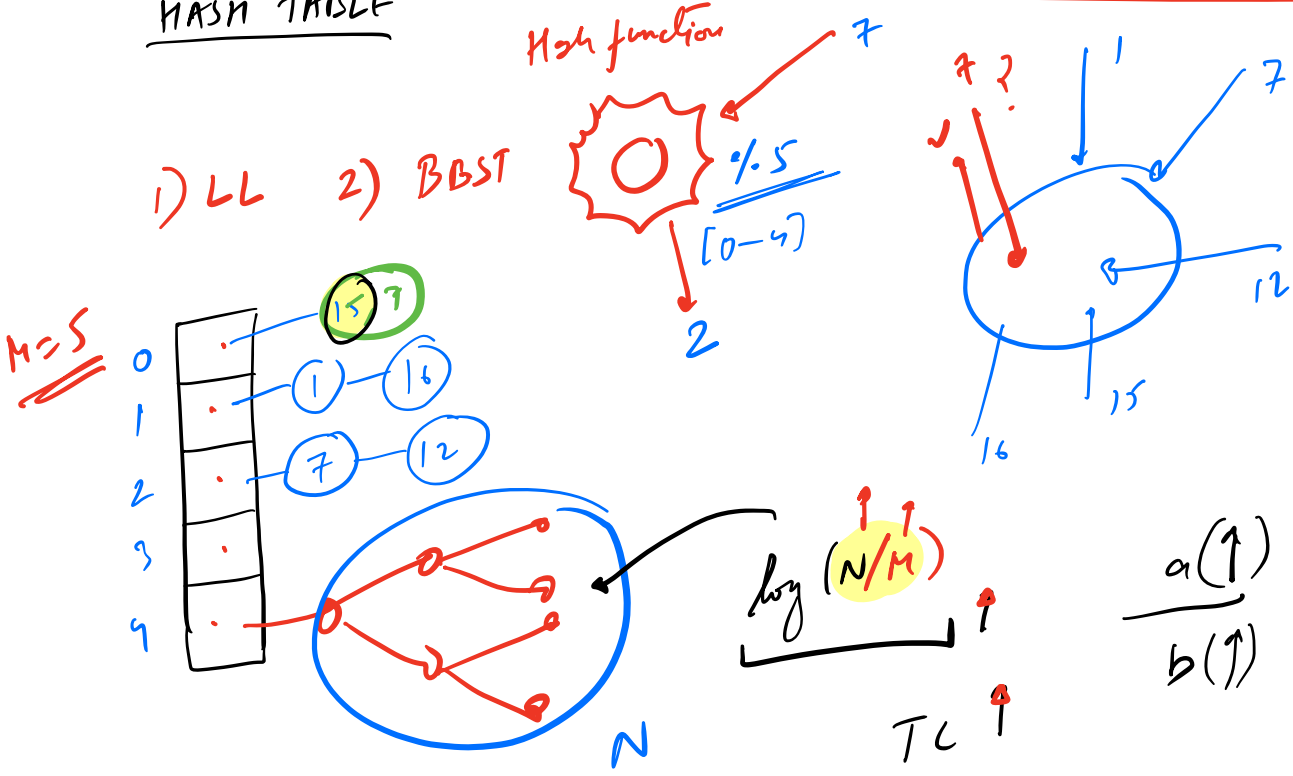




CHEMIST



HASH TABLE



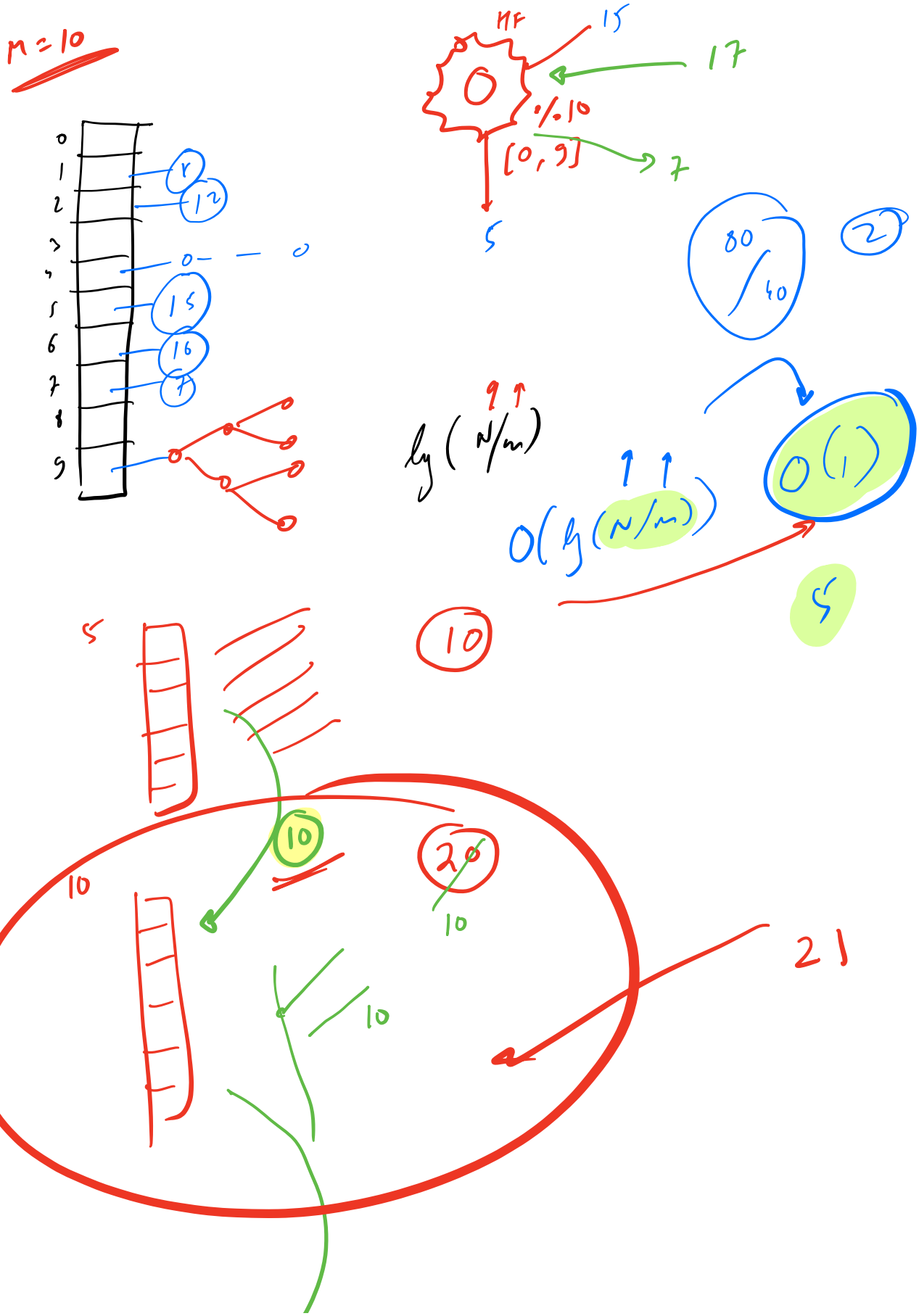
$\log\left(\frac{N}{M}\right)$

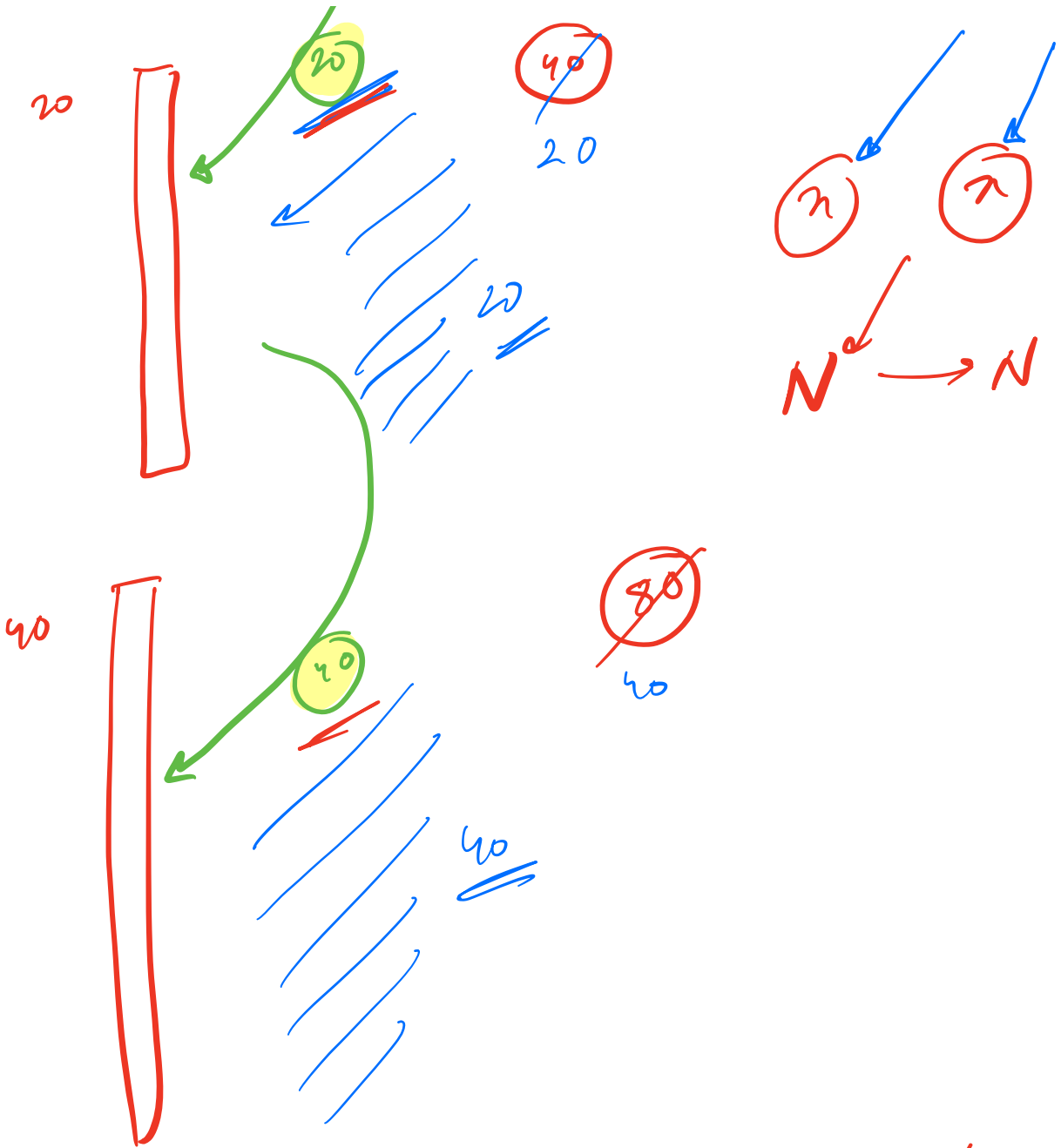
TC ↑

$\frac{a(\uparrow)}{b(\uparrow)}$

Load factor : (0.5)

RE-HASHING


m = 10
0
1
2
3
4
5
6
7
8
9
12
15
16
7
HF
15
17
0
%10
[0, 9]
7
5
80
40
2
lg (N/m)
O(1)
O(lg(N/m))
5
10
10
10
20
10
10
21

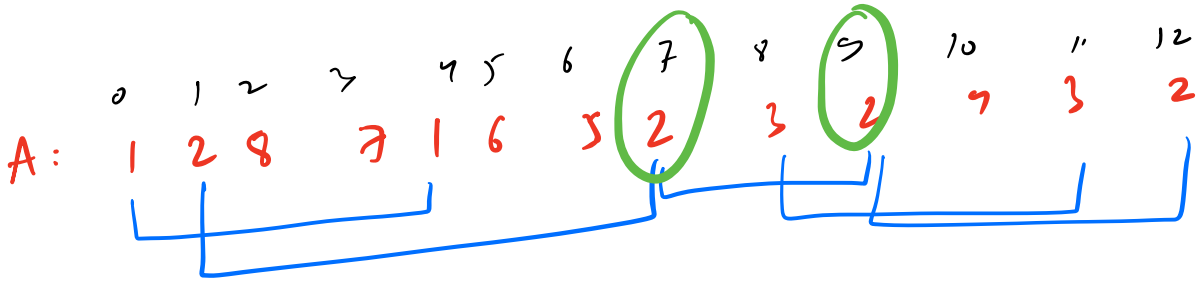


$$1 + 2 + 4 + 8 \longleftarrow + N/2$$

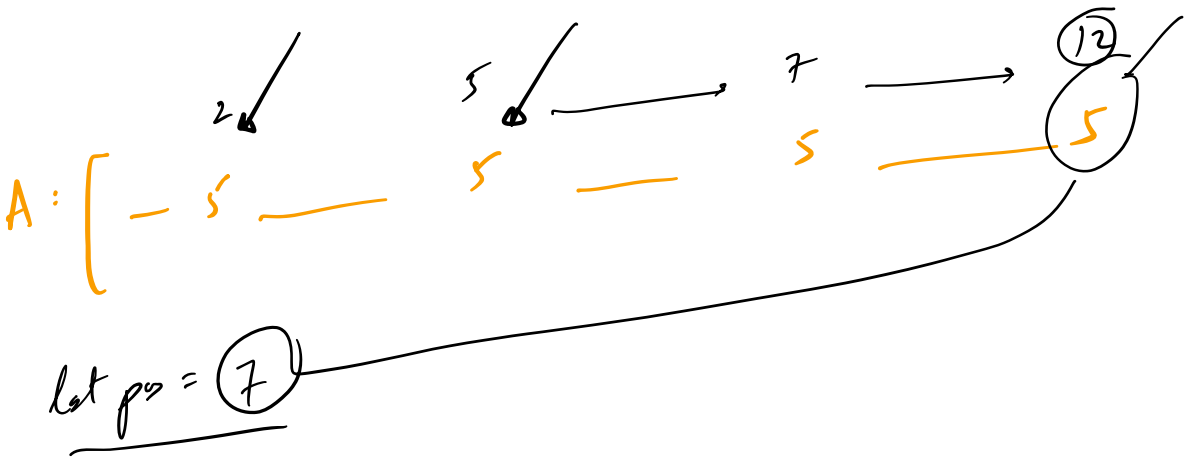
$$= N$$

Q Given an Array A[].

Find pair (i, j) : $A_i == A_j$ $(i \neq j$ & $abs(j-i)$ is MINIMUM



(7, 9)



/

	0	1	2	3	4	5	6	7	8	9	10	11	12
A:	1	2	8	7	1	6	5	2	3	2	7	3	2

HM [lastpos]

<1, 4>
<2, 7>
<8, 2>
<7, 3>
<6, 5>
<5, 6>
<3, 8>

ANS

∞
~~1 2~~
(i, j)
~~i, j~~
(0, 4)
(7, 9)

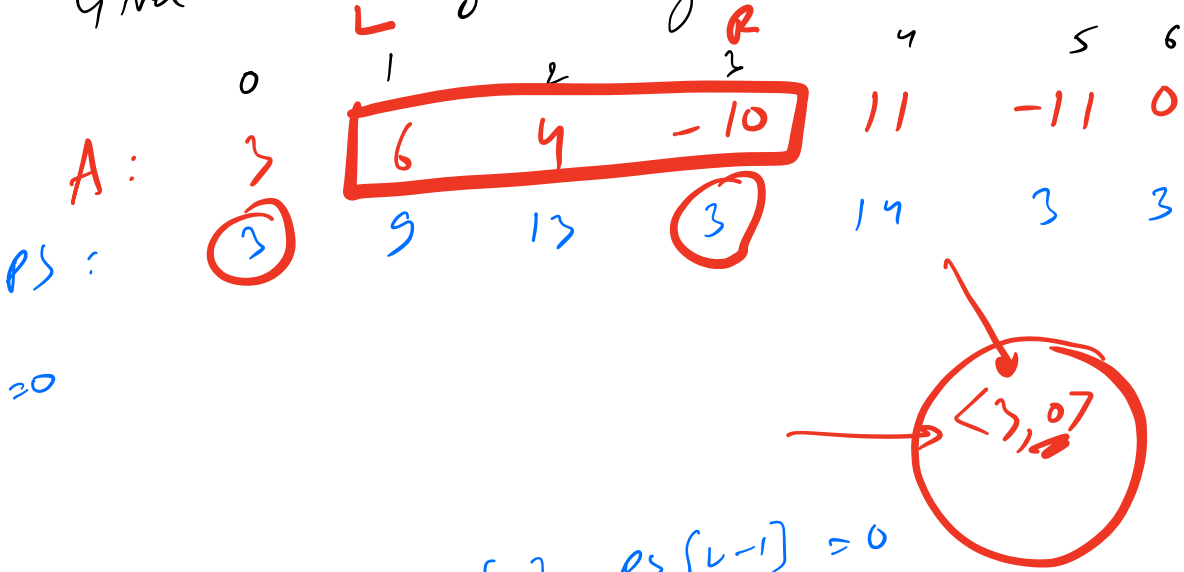
```
HashMap<int, int> hm;
ANS = ∞;   p = -1, q = -1;
for(i: 0 ⟶ N-1) {
    if(hm.contains(A[i])) {
        lastpos = hm[A[i]];
        diff = i - lastpos;
        if(diff < ANS) {
            ANS = diff;
            p = lastpos, q = i;
        }
```

TC = O(N)

SC = O(N)

```
    }
    hm[A[i]] = i;
}   ans (p, q);
```

Q Given an Arr; Longest S.A with SUM = 0

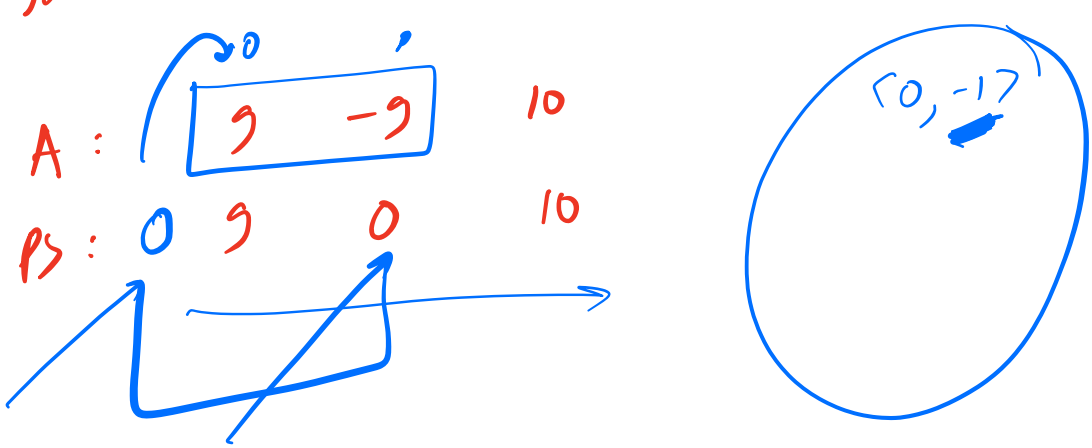


S=0

Sum [L, R] = PS[R] - PS[L-1] = 0

$\Rightarrow$ **PS[R] = PS[L-1]**

Idea: Store the 1st occ..



HashMap<int, int> hm;

hm[0] = -1;

ANS = -∞, p = -1, q = -1; ps = 0

for (i : 0 → N-1) { ps += A[i];

if (hm.contains(ps)) {

firstpos = hm[ps];

diff = i - firstpos;

if (diff > ANS) {

ANS = diff;

p = firstpos + 1, q = i;

}

}

else hm[ps] = i;

}

out(p, q);

TC = O(N)

SC = O(N)

Q Given an arry. Find the length of the longest consecutive sequence!

A: [100, 4, 200, 1, 3, 2] → 4

n, n+1, n+2, ...

A: [10 5 17 7 3 6 1 2 8] → 4

1) SORT

TC = O(N log N)

[1 2 3 5 6 7 8 10 14]

ANS ~~0~~ / 4

A: [10 5 17 7 3 6 1 2 8]

(5) (6) (7) (8) (N)

O(N)

(N)

n-1

n-1

0 1 2 3

4 5 6 7 8

9 10

13 14

TC = O(N)

SC = O(N)

Flip & find Nearest

Given a Binary Array of size N.

Q queries

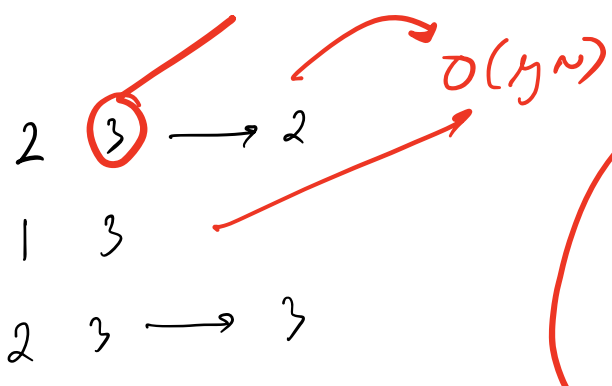
① 1 X

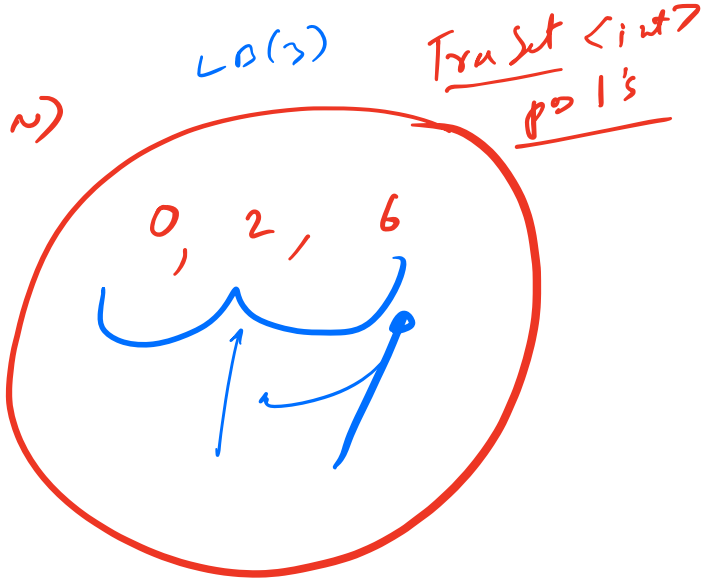
② 2 X

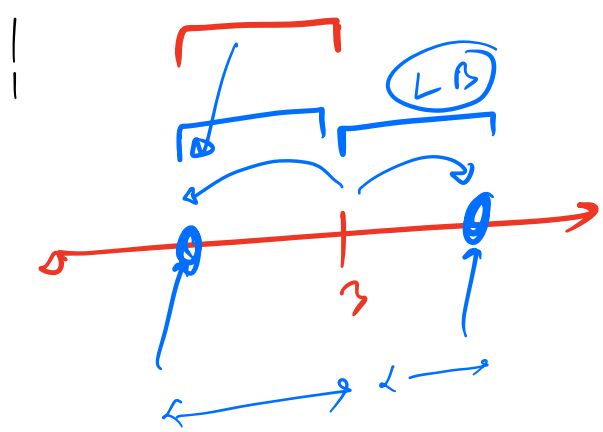
flip the bit at X^{th} pos

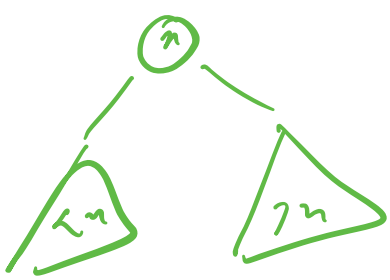
find the index of the nearest 1 from X.

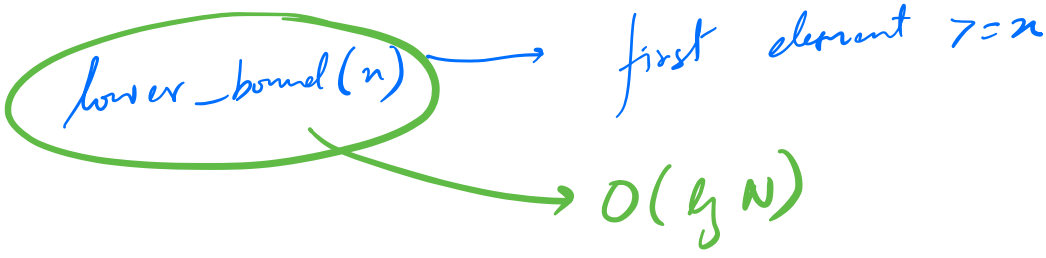
	0	1	2	3	4	5	6	7
A :	1	0	1	0	0	0	1	0











$$TC = O(N \lg N + Q \lg N)$$

lowerbound (n) → first no : >= x

Upperbound (n) → ——— : > x