NOTES:

HASHMAP

Good Evening

Today's Agenda

- → Hashmap Introduction
- → Hashset Introduction
- → Frequency of each query
- → First non repeating ele
- → distinct elements
- → Subarray with sum = 0 (Interview)
- → Story based question

* How things are implemented in HashMap ↓

Hashmap implementation

* Donot go for syntax, understand the logic

Hotel

Karan

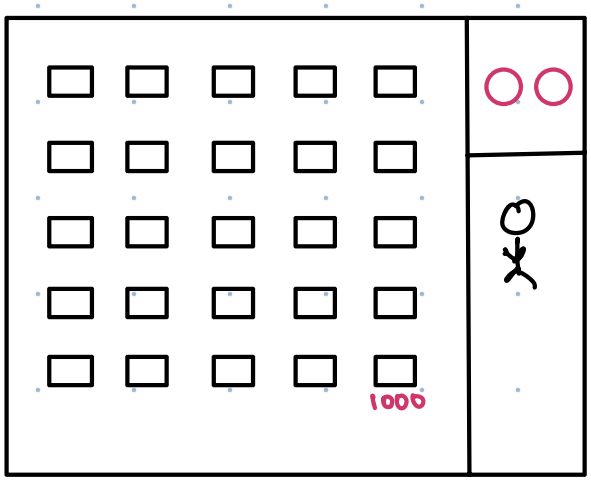
5 rooms

Manual Register

Room no.	Occupied
1	Yes
2	No
3	No
4	Yes
5	No

↓

1000 rooms



Room no. → {1, 1000}

boolean [] check = 1001

check [ ] =

True → Room is occupied

False → Room is available.

To check if a room is available or not → O(1)

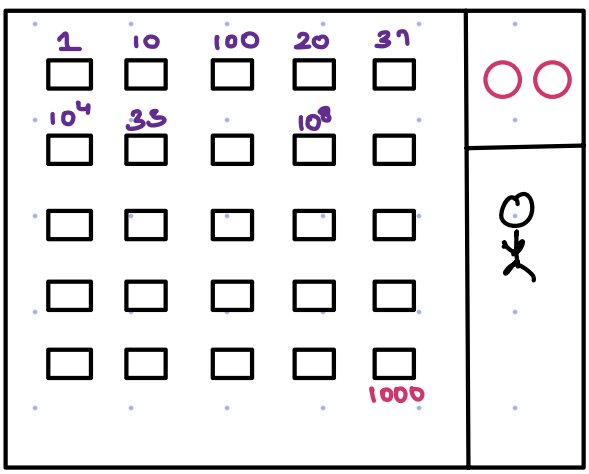
* Pandemic

Covid → Footfall has reduced drastically.

↓

Numerologist → { 1 to 10^9 }

1000 rooms



One to one mapping

boolean [ ] check = $10^9 + 1$

Advantage → Access the room status TC → O(1)

Issue → For 1000 rooms

↳ space of $10^9 + 1$

lot of unused memory in our array.

HashMap → A DS that holds the information in key-value pair.

HashMap < key, value >

Room no. | True/False

1	→	False
2	→	True
10	→	False
35	→	True
37	→	False

max size of HM = 1000

hm[2] = True

check if a particular room is available

TC: O(1)

→ Searching inside HM for a particular = O(1)

→ storing all values inside HM = O(n)

* Properties of HM

→ keys must be unique

→ value can be anything

→ HM doesn't maintain the order of insertion.

→ One null key is allowed

1. Population of every country

- India → 145
- USA → 65
- Russia → 14
- China → 147
- United Kingdom → 80

Key → Name of county
value → Population

HashMap < String, Long >

2. Number of states of every country

- India → 29
- USA → 50
- China → 25
- Russia → 21

Key → Name of County
value → No. of states

HashMap < String, Integer >

3. Name of all the states of every country

🇮🇳 India → Andhra Pradesh , Arunachal Pradesh , U.P , M.P , Karnataka ,....

🇺🇸 USA → New York , Washington , Texas ,...

🇷🇺 Russia → Moscow , Kazan , Samara ,...

🇨🇳 China → X , Y , Z ,...

Key → Name of country

Value → Name of all the states

HashMap < String , List < String >>

HashMap → To store key value pair

HashSet → To store unique keys

4. Population of each state in each country

🇮🇳 India	Karnataka → 50 Maharashtra → 72 Himachal Pradesh → 36
🇺🇸 USA	Texas → 18 Florida → 16 Washington → 32
🇷🇺 Russia	Kazan → 3 Samara → 5 Moscow → 7

Key = Name of country

Value = Population of each state

HashMap < String , HashMap < String, Integer >>

Hash - Map	Hash - Set
01. size() → No. of keys in HM	01. size() → No. of keys in HS
02. insert (key, value)	02. add (key)
03. search (key) → T/F	03. search (key) → T/F
04. remove (key)	04. remove (key)
05. get (key) → Fetch the value of this key	
06. update ↴ insert (key, value)	

A single operation in HM/HS → O(1)

	Java	C++	Python	JS	C#
Hash - Map	Hashmap	unordered_map	Dictionary	Map	Dictionary
Hash - Set	Hashset	unordered_set	set	set	Hashset

< Question > : Given N elements & Q queries. Find the frequency of elements provided in the query.

$(1 \le N \le 10^6)$

$(1 \le Q \le 10^5)$

N = 11

arr → [2 6 3 8 2 8 2 3 8 10 6]

0 1 2 3 4 5 6 7 8 9 10

Q = 4

2 → 3

8 → 3

3 → 2

5 → 0

BF Idea

For every query, iterate through the array & count the frequency.

TC : O(Q * N)

SC : O(1)

Idea -2

Use HashMap → key : Distinct ele

value : Count of distinct ele

arr → [2 6 3 8 2 8 2 3 8 10 6]

0 1 2 3 4 5 6 7 8 9 10

HM =

keys	value
2	3
6	2
3	2
8	3
10	1

Query

2 → hm.get(2) = 3

8 → hm.get(8) = 3

< / > Code

```
void printquery ( int [] A , int [] Q)
{
    HashMap < Int, Int > hm;
            // Iterate on A & build your HM
    for (i=0; i< n; i++){
        if (hm.search(A[i]) == true){
            int of = hm.get(A[i])
            int nf = of +1 ;
            hm.insert( A[i], nf );
        }
        else{
            hm.insert( A[i], 1 ) ;
        }
    }
    // Iterate on queries & print answer
    for ( i=0 ; i< Q.length ; i++){
        int ele = Q[i];
        if (hm.search(Q[i])= = true){
            print ( hm.get(ele))
        }
        else{
            print (0);
        }
    }
}
```

TC: O(N+Q)

SC : O(N)

10:21 pm → 10:31 pm

< Question > : Given N elements. Find the first non-repeating element.

N = 6 [1 2 3 1 2 5] Ans = 3

N = 8 [4 3 3 2 5 6 4 5] Ans = 2

N = 7 [2 6 8 4 7 2 9] Ans = 6

N = 4 { 1 2 2 4 2 } Ans = 1

BF Idea

For every element, iterate on array & look if its repeating or not.

TC : $O(n^2)$
SC : $O(1)$

if (count == 1) return A[i]

Idea -2

* Use HashMap

→ Populate the HM ⇒ key: Distinct ele, Value: Count

→ Traverse on original array

A[] = {2 6 8 4 7 2 9}

HM

{ 9 → 1, 7 → 1, 6 → 1, 2 → 2, 8 → 1, 4 → 1 }

< / > Code

```
int firstNonrepeating ( int [] A)
{
    HashMap < Int, Int > hm;
        // Iterate on A & build your HM
    for (i=0; i< n; i++){
        if ( hm.search(A[i]) == true){
            int of = hm.get(A[i])
            int nf = of +1 ;
            hm. insert ( A[i], nf );
        }
        else{
            hm. insert ( A[i], 1 ) ;
        }
    }

    for ( i=0; i<n; i++){
        if ( hm.get(A[i]) == 1 ) return A[i];
    }
    return -1;
}
```

Tc : O(n)

Sc : O(n)

< Question > : Count of distinct elements

N = 5 [3 5 6 5 4] Ans = 4

N = 3 [3 3 3] Ans = 1

N = 5 [1 1 1 2 2] Ans = 2

Idea → Insert all elements inside hashset & it will not store distinct elements.

< / > Code

```
Hashset < Integer > set

for (i=0; i<n; i++){
    set.add(A[i]);
}
return set.size();
```

TC : O(n)
SC : O(n)

< Question > : Given an array of N elements. Check if there exists a subarray with sum equal to 0.

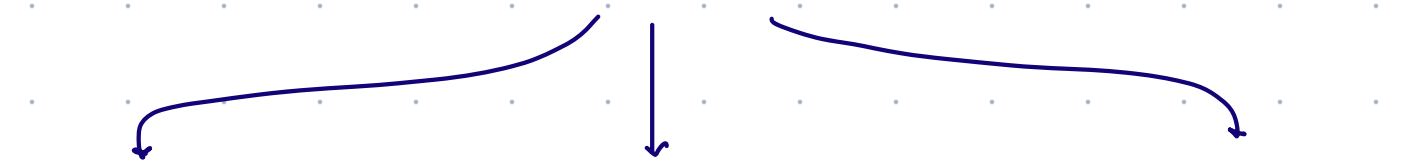
$(1 \le N \le 10^5)$

N = 11 arr → [2 2 1 -3 4 3 1 -2 -3 2] Ans = true

0 1 2 3 4 5 6 7 8 9

BF Idea

Get all the subarrays & check if the sum = 0 or not



3 nested loops

TC : $O(n^3)$
SC : $O(1)$

Use prefix loops

TC : $O(n^2)$
SC : $O(n)$

Carry forward

TC : $O(n^2)$
SC : $O(1)$

Idea -2

N = 10

	0	1	2	3	4	5	6	7	8	9
arr →	2	2	1	-3	4	3	1	-2	-3	2
psum[ ] →	2	4	5	2	6	9	10	8	5	7

$sum[i \; j] = psum[j] - psum[i-1] = 0$

$psum[j] = psum[i-1]$

$pf[2] = 5 \qquad pf[8] = 5$

$sum[3-8] = pf[8] - pf[2] = 0$

* Conclusion → Repeating values in psum array denotes subarray sum = 0

	0	1	2	3
arr →	-2	-1	3	5
psum[ ] →	-2	-3	0	5

HS: { 0, -2, -3 }

Note → If 0 is present in psum[ ], we have a subarr with sum = 0

- → If (psum[i] == 0) return true :
- → Explicitly add a 0.

< / > Code

```
boolean subarrayZero (int [] A){

    int pf[n] ⟶ {TODO}

    HashSet<Integer> set

    set.add(0);

    for (i=0; i<n; i++){

        if ( set.search (pf[i]) == true) return true;

        else  set.add ( pf [i]);

    }

    return false;

}
```

TC : O(n)

SC : O(n)

* Use carry forward approach to save prefix array.

Scenerio

SCALER has a series of contests to help learners improve their coding skills. Each learner tries these contests to practice, but not everyone participates the same amount. **SCALER** wants to know which learner is participating the least so they can help them more or figure out how to get them more involved.

Problem

You have a list of id where each id represents a learner who tried a contest. A learner's number shows up more times if they try more contests. Your job is to write a program that looks at this list and finds out who has tried the **least number of contests**.

Example

Input:

learner_ids = [101, 102, 103, 101, 102, 101, 104, 105, 106, 105, 105]

106 → 1
101 → 3
102 → 2
103 → 1
104 → 1
105 → 3

Output = learner with least no. of participation

= {103, 104, 106}

A[] = {3 2 1 1 3 7}

HM

{ 3 → 2
2 → 1
1 → 2
7 → 1 }

least value = 1

Ans = {2, 7}

* Learn how to iterate on HM

HW

* Count subarray with sum = 0
* Longest subarray with sum = 0