8th December : Recursion , Maths & OOPS

↳ 2nd Dec → PS session
7th Dec → PS Session

Contest 1 Reattempt 3
↳ 5th December.

Revision : ~~Re Watch lecture Recording~~

~~Go through lecture notes.~~

1) Solve unseen problems.
↳ Pending assig.
↳ Pending add. problem
↳ LeetCode, InterviewBit, GfG, HR ......

Able to solve

Not able to solve (Revise)

1) Solution
2) Lecture Notes
3) Recording (2x speed)

Medium

Hashing : Introduction

↳ Int: Hashing.

2) Problems

3) Internal Implementation

HM/HS ⇒ Data Structure

Impl.

Hashing ⇒ Concept

Hash Map

Data Structure which implements a Hash Table.

Eg

Key Student_email	Value. PSP
ayush.sharma@scaler.com	100
parththerool@gmail.com	100

awesomeguyharshit@gm	98
bipindcool22@gmail.com	35
hackerguy@gmail	56
⋮	⋮

List of < Key , Value > pair.

Key $\longrightarrow$ (Hash function) $\longrightarrow$ where to store value for the Key

$$\begin{tikzcd}
-\infty \arrow[dr] & & a \arrow[dd, leftrightarrow] \\
& H(x) \arrow[ur, bend left] \arrow[dr, bend right] & \\
\infty \arrow[ur] & & b
\end{tikzcd}$$

<u>Modulo</u> $\Rightarrow$ $\underset{\substack{\downarrow \\ [-\infty,\, \infty] \\ \text{Range}}}{x} \,\%\, M = [0, M-1]$

T.C. of Search, insert, deletion $= O(1)$

Cond:

1) Key must be unique.

2) Values can be anything.

No → Ass / H.W.

Advanced batch → Add. problems

HashMap < $\underset{\text{Key datatype}}{\underline{\qquad\qquad}}$, $\underset{\text{Value datatype}}{\underline{\qquad\qquad}}$ > name ;

Q) $\underset{}{\overset{\text{Value}}{\underline{\text{Population}}}}$ of every $\overset{\text{Key}}{\underline{\text{country}}}$

HashMap < String, Long >

Q) $\overset{\text{Value.}}{\underline{\text{No. of states}}}$ for every $\overset{\text{Key}}{\underline{\text{country}}}$

HashMap < String, int >

Q= Name of all states for every country.

(Value over "Name of all states", Key over "country")

~~< Key , Value >~~
~~India , UP~~
~~India , Rajasthan~~
~~India , MP~~

< Key , Value >
India , [UP, Rajas., MP....]

HashMap < String, List <String>>

Q= Population of Each State for every Country

(Value over "Population", Key over "Each State", Key over "Country")

HashMap < String, HashMap < String, Long>>

HashMap

Java	C++	Python	JS	C#
HashMap	unordered_map	dictionary	map	dictionary

Hash Set

Java	C++	Python	JS	C#
HashSet	unordered_set	set	set	HashSet.

Q1 Given N elements & Q no. of queries.

Query : Given x ⇒ Return the frequency of x in the array.

Eg:

$$A = \begin{array}{cccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 2, & 6, & 3, & 8, & 2, & 8, & 3, & 8, & 10, & 6 \end{array}$$

Q = 3

x	freq of x.
2	2
8	3
5	0

Solⁿ 1) Brute Force

⇒ $\underbrace{\forall \text{query}}_{Q}$ ⇒ $\underbrace{\text{Iterate the array}}_{N}$ & find freq.

T.C. = O(Q×N)
S.C. = O(1)

Q How can we improve T.C. ??

Key Value

< Element , frequency >

$$\begin{array}{cccccccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ A = [& 2, & 6, & 3, & 8, & 2, & 8, & 3, & 8, & 10, & 6] \\ & & & & & & & & & & i \end{array}$$

< Key	Value >
2	~~1~~ 2
6	~~1~~ 2
3	~~1~~ 2
8	~~1~~ ~~2~~ 3
10	1

$\Big\} O(N)$

$Q = 3$

Elements	Frequency	
2	2	$\Rightarrow O(1)$
8	3	$\Rightarrow O(1)$
5	0	$\Rightarrow O(1)$

$\Big\} Q$

$T.C. = O(N+Q)$

Hash Map

1) Insert (Key, Value) : Insert a new entry
2) Containskey (Key) : Returns True if the given Key is present in HM.
3) update (Key, updated value) : Update the value for given Key
4) size() : Return the size of the HM.

$\Downarrow$

No. of entries

$\Downarrow$

No. of Keys.

5) delete (Key) : Delete the entry for given Key.

HashSet

1) Insert (Key)
2) Containskey (Key) : Returns True if the given Key is present in HS
3) Size() : Total no. of Keys
4) delete (Key) : Removes the given Key from HS

Code

1) Create a frequency map (HM)

```
HashMap < int , int > hm;

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

    if (! hm.containsKey (A[i])) {
        hm.insert ( A[i], 1);
    }
    else {
        hm.update (A[i], hm.get(A[i])+1);
    }
}
```

2) Answer all queries   // queries [N]

```
for (i=0; i<Q; i++) {

    if (hm.containsKey (queries[i]) {
        print (hm.get (queries [i]);
    }
    else {
        print (0);
    }
}
```

Q Given an integer array of size N.
Return the first non repeating element

Eg $N = 6$

$A = [1, 2, \underline{3}, 1, 2, 5]$

Ans = 3

$N = 8$

$A = [4, 3, 3, 2, 5, 6, 8, 5]$

Ans = 4

Sol^n I) Brute Force

$\forall_i \Rightarrow$ Iterate & find freq. of $A[i]$ } HM

$\Downarrow$

The first i for which freq of $A[i]$ is 1 becomes my answer.

2) Opt.

1) Create a frequency map (HM)

2) Iterate over the array & check the freq of each ele.

3) The first ele with freq 1 is ans.

Q In point 2, can we iterate over HM

NO

The elements of HM are not stored in the order of insertion.

Q Given an integer array of size N. Return the count of distinct elements in the array.

$N = 5$,

$A = [3, 5, 6, 5, 4]$

Ans = 4

Sol^n Use HashSet

1) Insert all elements in a HashSet
2) Return its size.

Code

```
HashSet <int> hs;

for (i=0; i<N; i++) {
        hs.insert (A[i]);
}

return  hs.size();
```

T.C. = O(N)
S.C. = O(N)

A = [ ]   ~~Range = [0, 10^9]~~

Range [0, 100]   Size = [10^4]