

Content →

- Hashing
- Distinct elements
- Frequency of an element
- First non repeating
- check subarray with sum=0

Hashing

Hotel

10 rooms

register

1000 rooms

bool ch[1000]

10^9 rooms

bool ch[10^9]

5000 rooms occupied

X

How to not waste space?

Hashmap

{ } --
 < 20, Ayush >
 < 15, Aman >
 < 25, Naman >

Key-value pair
Will one key
have only one
entry

Keys are unique

Hashset

15
20
22

Only key

Keys are unique

1) Store population of every country

key, value
↓ ↓
string long
HashMap < string, long > hm

2) For every country, number of states

HashMap < string, int > hm

3) For every country, store all state names & state population

key value
string hashmap < string, long >

India, { UP: 300
Rg: 250
Guj: 200 }

Note: Value can be ANYTHING (any data type)
Key: { string, int, long, float,
List, char }

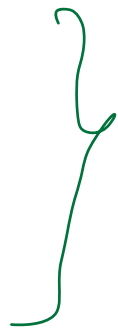
HashMap < key, value >
insert (key, value)
search (key)
remove (key)
size



All operations
are $O(1)$

Can key have multiple values NO

HashSet < key >
insert (key)
remove (key)
search (key)
size



All operations
are $O(1)$

Q1) Given N array elem, find no of distinct elements

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

Idea: Use HashSet

obs: Size of HashSet is our answer

3
5
6
4

```
int unique (int arr[]) {  
    HashSet<int> hs  
    for (i=0; i<n; i++) {  
        hs.insert(arr[i])  
    }  
    return hs.size()  
}
```

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

HashMap < String, String> hm = new

HashMap < String, String> ()

Q2 Find frequency of numbers
Given N numbers & Q queries, for each query find frequency of that number

Eg: $A[10] = [2, 6, 3, 8, 2, 8, 2, 3, 8, 10]$
 $Q[4] = [2, 8, 3, 5]$
 3 3 2 0

Hashmap $\langle \text{elem}, \text{freq} \rangle$

2: $\text{hm.get}(2)$
8: $\text{hm.get}(8)$
3: $\text{hm.get}(3)$
5: 0

HM	
2: +2	3
6: 1	
3: +2	
8: +2	3
10: 1	

Code

```
void frequency (int arr[], int Q[]) {
    hashmap <int, int> hm
    for (i=0; i<n; i++) {
        if (hm.search(arr[i]) == true)
            hm[arr[i]]++
        else
            hm.insert(arr[i], 1)
    }
    for (i=0; i<q; i++) {
        if (hm.search(Q[i]) == true)
            print(hm.get(Q[i]))
        else
            print(0)
    }
}
```

TC: $O(n+q)$
SC: $O(n)$

Q3 Find first non repeating elem
wrt array

eg 1 $A[6] = [1, 2, 3, 1, 2, 5]$ ans = 3
 $A[8] = [4, 3, 3, 2, 5, 6, 4, 5]$ ans = 2

Idea: 1) Create freq hashmap
 2) Iterate on array
 return the first elem
 freq = 1

Code

```
int nonRepeat (int arr[]) {  
    1) create frequency hashmap  
    for (i=0 ; i<n ; i++) {  
        if (hm.get(arr[i]) == 1)  
            return arr[i]  
    }  
}
```

TC: $O(N)$ SC: $O(N)$

Q4 Check if there exist subarray with sum = 0

Eg1 $A[7] = [2, 3, -1, 4, -3, 10, 4]$ true

Eg2 $A[5] = [1, 2, -1, -2, 4]$ true

Idea: PF sum of subarray $[s:e] = pf[e] - pf[s-1]$

we want this to be 0.

$$pf[e] = pf[s-1]$$

Obs: 1) If any pf entry repeats OR
2) If any pf entry = 0

TC: $O(N)$ SC: $O(N)$

Code

```
bool subzero (int arr[]) {  
    1) Create pf array
```

```
    2) Create freq hashmap on pf array
```

```
    a) if (hm.search(0) == true) {  
        return true
```

```
    b) for (int i=0 ; i<n ; i++) {  
        if (hm.get(pf[i]) > 1)  
            return true
```

```
    c) return false
```


$\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ [2, 3, -1, 4, -3, 10, 4] \\ 2 & 5 & 4 & 8 & 5 & 15 & 19 \end{array}$

$\begin{array}{ccccc} 0 & 1 & 2 & 3 & 4 \\ [1, 2, -1, -2, 4] \\ 1 & 3 & 2 & 0 & 4 \end{array}$

\Rightarrow as 0 is present in pf, true

$O: i$

$\Rightarrow \emptyset$

{done}

iterators.

$\begin{array}{c} * \\ * * \\ * * * \\ * * \\ * \end{array}$