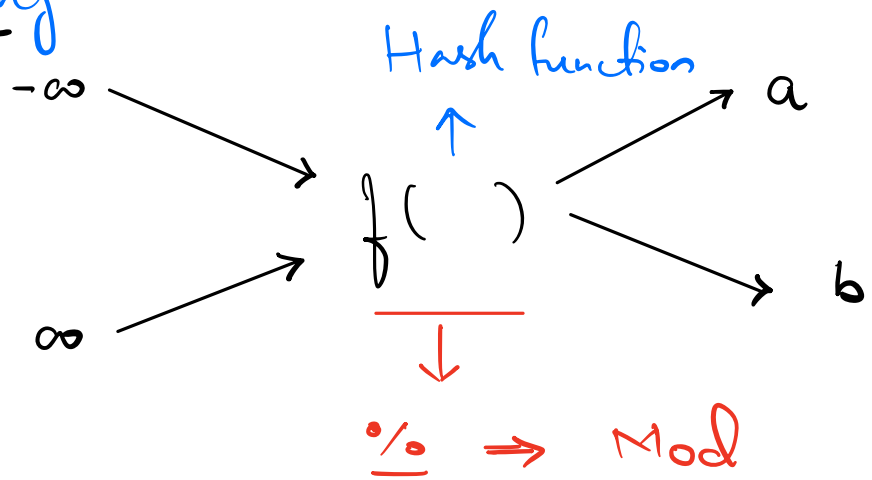
HashMap & HashSet

Hashing



$$\underset{[-\infty,\ \infty]}{x} \ \% \ M = [0, M-1]$$

Chikamangalur

↳ Checkin to Radisson (30 rooms)

Hash Table

Key: Room No.	Value: Occupied
101	✓
102	✓
103	✗
104	✓
⋮	⋮

HashMap < Key, Value > ⇒ Collection of Key-Value pairs.

☆ Keys in a HashMap are unique.

Searching for a Key is O(1) T.C.

HashMap < String, int >

Q. Population of every country

Value Key

HashMap < String, Long > populationByCountry.

Q. No. of states for every country

Value Key

HashMap < String, int >

Q. Name of all States of every country

Value ⇓ List < String >

Key

HashMap < String, List < String >> ...

Q. Population of Each State in every Country

Value Value Key

Key.

HashMap < String, HashMap < String, Long >>

Hash Set

⇒ Only unique Keys (No value)

	Java	C++	Python	JS	C #
HashMap	HashMap	unordered_map	dictionary	map	dictionary
HashSet	HashSet	unordered_set	set	set	HashSet

HashMap functions

1) Insert (Key , Value)
   ⇒ Inserts a new Key-Value pair in H.M.
   (If Key is already present, no change)

2) Delete (Key)
   ⇒ Deletes the Key-Value pair with this Key

3) Get (Key)
   ⇒ Return the value for the given Key

4) containsKey (Key) Search
   ⇒ Returns True if the given Key is present, false otherwise.

5) size ()
   ⇒ No. of Keys present in HashMap.

6) Update (Key , Value)
   ⇒ Update the value of Key if present.

HashSet

1) Insert (Key) ⇒ Inserts a new Key.
If already present, no change

2) size() ⇒ No. of Keys in HashSet

3) Search / containsKey (Key)
⇒ Returns True if a Key is present, false otherwise

4) Delete (Key) ⇒ Removes the given Key if present.

T.C. for all above functions = O(1)

Given an integer array of size N.
Q queries ⇒ find the freq of a given element.

N = 11

A = { 2, 6, 3, 8, 2, 8, 2, 3, 8, 10, 6 }

Q = 4

Element		Frequency
4	⇒	0
2	⇒	3
3	⇒	2
8	⇒	3

1) Brute force

$\forall$ query $\Rightarrow$ Iterate & count freq of element

$\Downarrow$ $\Downarrow$

$Q \times N = O(Q \times N) \Rightarrow$ T.C.

S.C. $= O(1)$

Q How to improve T.C..

Create a HashMap to store the freq of every element.

$N = 11$

	0	1	2	3	4	5	6	7	8	9	10
A =	2	6	3	8	2	8	2	3	8	10	6

HM

int Element	int freq
2	~~1~~ ~~2~~ 3
6	~~1~~ 2
3	~~1~~ 2
8	~~1~~ ~~2~~ 3
10	1

Code

```
void printFrequency (Q[], A[]) {
    HashMap <int, int> freqmap;
    q = Q.length
    N = A.length

    for (i=0; i<N; i++) {          // O(N)
        if (! freqmap.containsKey (A[i])) {
            freqMap.insert (A[i], 1);
        }
        else {
            freq = freqMap.get (A[i]);
            freqMap.update (A[i], freq+1);
        }
    }
    for (i=0; i<q; i++) {          // O(Q)
        if (! freqMap.contains (Q[i])) {
            print (0);
        }
        else {
            print( freqMap.get (Q[i]);
        }
    }
}
```

T.C. = $O(N+8)$
S.C. = $O(N)$

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

N = 7

A = { 1, 2, 3, 1, 4, 2, 5 }

→ Ans

A = { 4, 3, 3, 2, 5, 1, 6, 4, 5 }

Ans

Sol^n

⇒ Create a freq map of all elements of array.

H.M. is unordered.

~~⇒ Iterate over H.M. (freq map) & return the first element with freq = 1.~~

⇒ Iterate over array & find the first element whose freq = 1

T.C. = $O(N)$

S.C. = O(N)

Q Given an integer array of size N. Find count of distinct elements.

A = { 3, 5, 6, 5, 4 }

Ans = 4

A = { 3, 3, 3 }

Ans = 1

Sol^n Iterate & insert all elements to Hashset

Return size of Hashset.

Code

```
int countDistinct ( A[] ) {
    HashSet <int> hs;
    for (i=0; i< A.length; i++) {
            hs.insert (A[i]);
    }
    return hs.size();
}
```

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

Given an array of size N.
Check if there exists a subarray whose sum = 0;

	0	1	2	3	4	5	6	7	8	9
A =	2,	2,	1,	-3,	4,	3,	1,	-2,	-3,	2

True

Sol^n 1) Brute force

$\Rightarrow$ $\forall$ subarray $\Rightarrow$ Calculate the sum & check if it is 0.

T.C. = $O(N^3)$

$\downarrow$ P.S. / Carry forward

$O(N^2)$

$\downarrow$?? Can we optimise ??

2) Prefix Sum

	0	1	2	3	4	5	6	7	8	9
A =	2,	2,	1,	-3,	4,	3,	1,	-2,	-3,	2

PS: {2, 4, 5, 2, 6, 9, 10, 8, 5, 7}

$$Sum[i, j] = 0$$

$$\Downarrow$$

$$PS[j] - PS[i-1] = 0$$

$$PS[j] = PS[i-1] \Rightarrow$$

```
if (i == 0) {
        PS[j] = 0;
}
```

$$A = \{4, 2, -6, 3\}$$

$$PS = \{4, 6, 0, 3\}$$

Code

```
Prefix[N];
Prefix[0] = A[0];
if (Prefix[0] == 0) { return True; }
for (i = 1, i < N; i++) {
        Prefix[i] = Prefix[i-1] + A[i];
        if (Prefix[i] == 0) {
                return True;
        }
}
```

```
HashSet<int> hs;
for (i = 0; i < N; i++) {
        if (hs.containsKey(Prefix[i])) {
                return true;
        }
        hs.insert(Prefix[i]);
}
```

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

Q Count the no. of subarrays with sum = 0.

ans % (10^9+7)

	0	1	2	3	4	5	6	7	8	9	10	11
A =	2,	2,	1,	-3,	4,	3,	1,	-2,	-3,	-3,	2	-2
PS:	2,	4,	5,	2,	6,	9,	10,	8,	5,	2	5	2

(1, 3)
(1, 9)
(4, 9)

(3, 8)
(3, 10)
(9, 10)

(1, 11)
(4, 11)
(9, 11)

Bit Manipulation