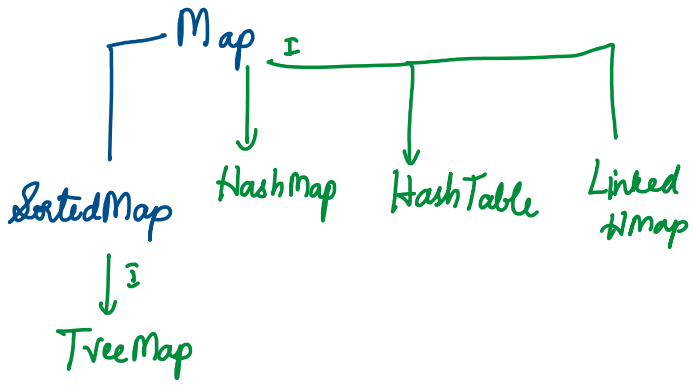
HashMap Internal Working

29 September 2024 23:19



① Why is Map not under Collection

Because it is not a collection of single value.

It deals with key value pair.

Map - Its an interface

Implementations : HashMap
HashTable
LinkedHMap
TreeMap

Object maps keys to values

No duplicate keys

② Override existing keys

Map - methods

size()

isEmpty

containsKey

containsValue

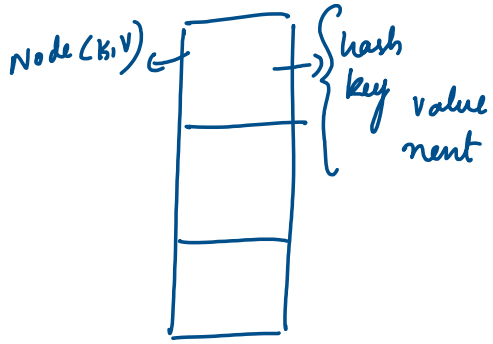
get
put
remove

Map Internal Storage

Entry → subinterface

Map.Entry<K,V>

In hashmap we have an array of Entry<K,V>



i) During put op in Hmap

put(1, "Sudeep");

↓

1) hash(key)

1 → compute hash → 1234567

1234567 % size of table

For Eg:- 1234567 % 3

↓

1 → index

So store it in index 1

Collision theory

put(10, "15");

10 Hash → 515100 % 3

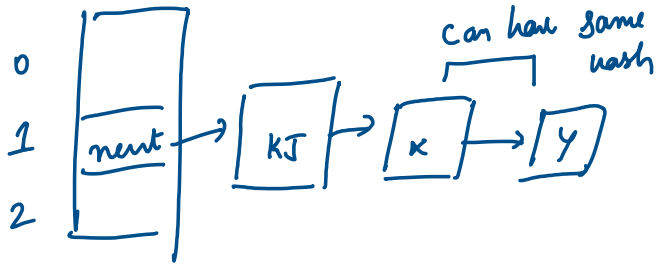
1

1 → Index

But Index 1 is already filled.

i) So it checks if same key.

ii) So it places in the next of index 1 node.



Get in HMap

get(5) → Hash → 815100

815100 % size of table.

Index → x

At this index it will traverse through the LL & check the key value.

---

* Contract b/w HashCode & equals

i) if Obj1 == Obj2 are equal then their Hash should be same

ii) if 2 objects hash is same then it doesn't mean that Obj1 == Obj2

How HashMap maintain O(1)

on Insertion, Deletion. etc.

Load Factor = 0.15

Initial Capacity = 16

So $\frac{3}{4} \times 16 = 12$

Once it reaches 12 it will re hash.

It will increase size (doubles).

1) Lower load factor → lesser collisions but more resize.
2) Higher load factor → resize less often but more collisions.

TREEFY Threshold

It is limit of linked list to be created / added in an array during collision

Once threshold is reached LL is converted to tree → BST.

Bucket / Bin Count → size of LL.

Time Complexity → $O(n)$

Tree → $O(\log n)$