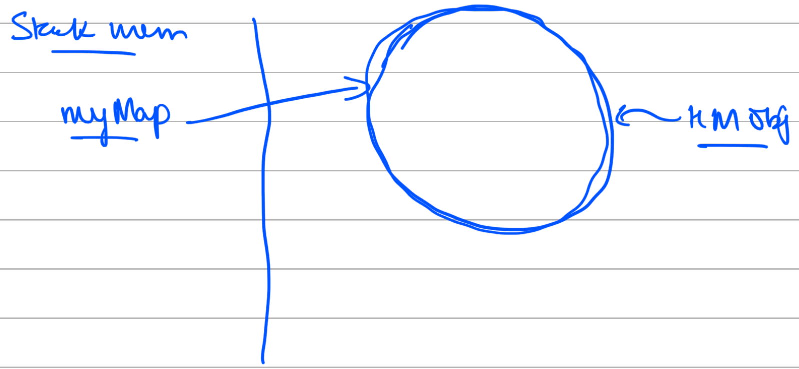
HASHMAP

@thriverashish (ASHISH GUPTA)

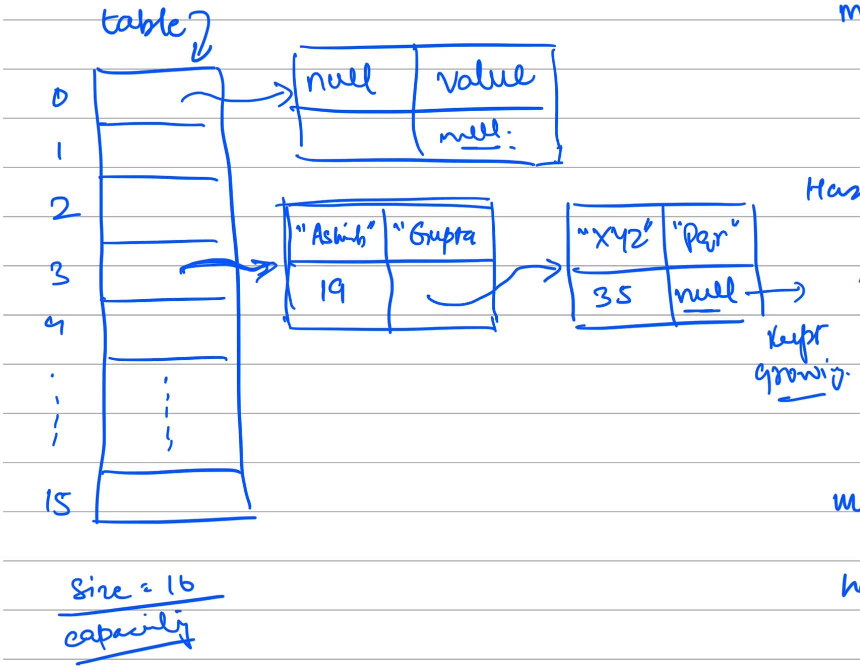
Map < String , String > myMap = new HashMap<>();



$$\underbrace{get(\_)\ ,\quad put(key, value)}_{O(1)}$$

table ← array of type Node → inner class which extends Map.Entry

transient Node[] table



Case 1 : Key is not null.

mymap.put("Ashish", "Gupta");

↑ Key.

Hash of Ashish is 19

19 % capacity

19 % 16 = 3

mymap.put("XYZ", "Pqr")

hash of XYZ = 35

35 % 16 = 3

↗ index.

Node — {

Key
Value
hash

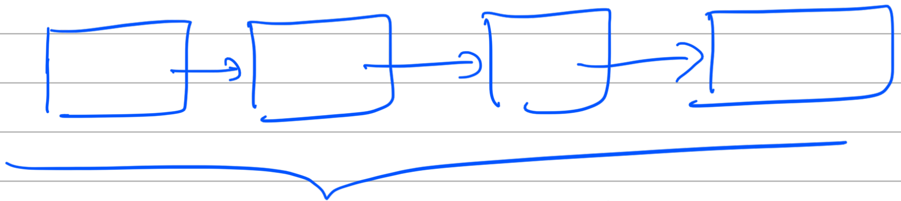
Node next

}

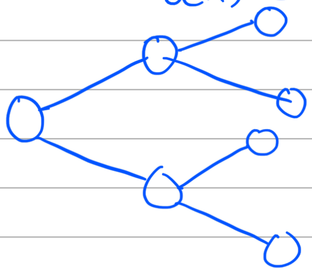
Case 2 : Key is null

If size of LL is grown upto Threshold. →

from Java 8 → collision is further handled via Self Balancing Binary Search Tree



$O(K) \simeq O(n)$



Self Bal BST

$O(\log n)$

Tree.Node

loadfactor → 75% By default

If more and more elements are inserted
⇒ increases collision.

capacity of HM → 16 loadfactor 75%

75%

12 → Size of HM. is

capacity of HM will increase.

newCapacity = Oldcapacity << 1

16 << 1

32

How 16 rep in Binary = _ _ _ _ _ 10000 = 16

2^4 2^3 2^2 2^1 2^0

left Shift by 1

newCap. $\ldots 100000 = \boxed{32}$

$2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

Max Capacity = $1 << 30$

Get Operations

equals given key n present key.

key is null

table[0]

key is not null.

- cal Hashcode of key.
- maps index & search.
- iterates over table[index]

Checks hashcode & equals

if matches return value.

else null. (return).

@thriverashish

Road map.

insta

twitter

linkedIn.

Ashish