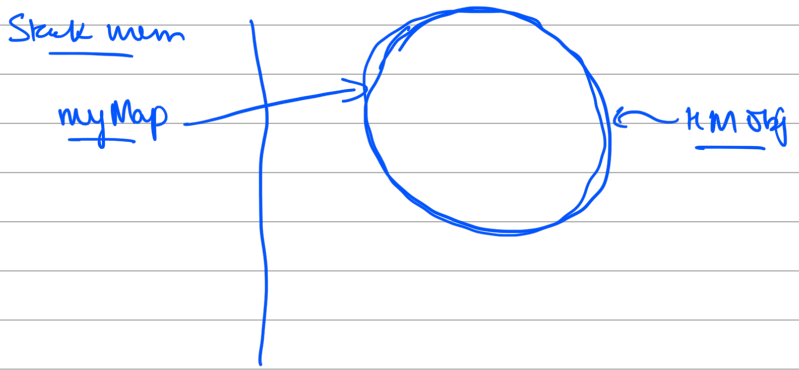
HASHMAP

@thriverashish (ASHISH GUPTA)

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



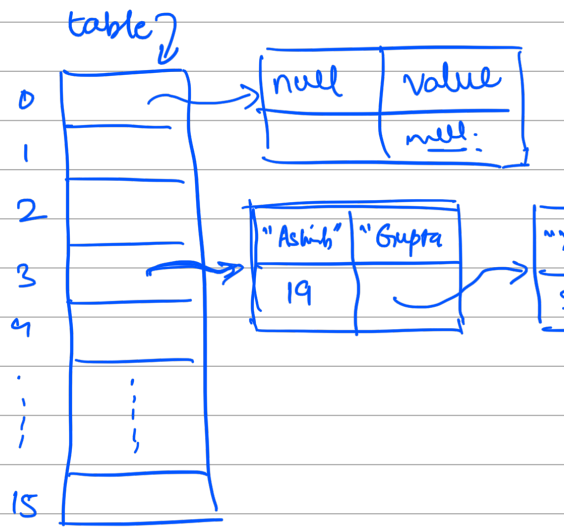
get(_) , put(key, value)

O(1)

table ← array of type Node

Node → inner class which extends Map.Entry

transient Node[] table



Size = 16
capacity

Case 1: Key is not null.

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

Key.

Hash of Ashish is 19

19% capacity

keep growing.

19 % 16 = 3

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

hash of XYZ = 35

35 % 16 = 3

index.

Node ____ {

- Key
- Value
- hash

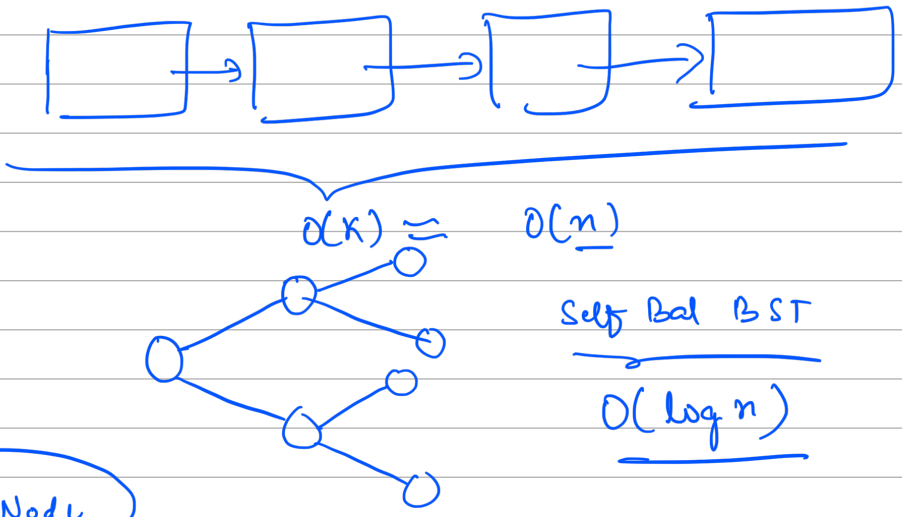
Node (Next)

}

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) ≃ 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 up in Binary = ...10000 = 16

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

left shift by 1

newCap. ....100000 = 32
$2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

Max Capacity = 1 << 30

Get Operation

Key is null
table[0]

Key is not null.
- Call HashCode of key.
- maps index & search.
- Iterates over table[index]
checks hashCode & equals
if matches return value.
else null. (return).

equals from given key

@thriverashish
insta ← twitter ← linkedIn. ← roadmap.

Ashish