Hashset & Hash Map

Hashmap 哈希表/散列表

HashMap<key, value>

1. 需求 look up → HashMap can solve

key = Integer/long/String

value = Student class

key = Integer

value = String

list<String/Integer>

a) 查找：

* key 最好是单值 → 简单表格

① 逐一查找 // O(N) brute-force 暴力法

② Binary-Search // O(lgN)

③ Hashmap O(1)

(Bucket)[]

* Array：通过 index 查找对应 element O(1)

⇒ Get

value

key → index 希望 O(1) → Bucket → 在指定 Bucket 找 key

<key, value>

key 的 Type 不定

could be int, String, Myclass

O(1)

privacee int hashFunc

* key → index

default behavior

定义任何方 class 都是 Object 的子类会

① class Object { 实体 method class Myclass { 继承 Object

public int hashcode(){ ..... }

customize hashcode

@Override

public int hashcode(){ ..... } → 自定义

生成 index

☆ // 这里类

② private int hasFunc(K key){ → 作成 hashmap 的 key

if(key==null) return 0;

return Math.abs(key.hashCode() % buckets.size());

↳ int % buckets.size → 得到一个可以直接 apply 到 buckets 的 index

* 为什么一定要大写：

因为要把 primitive 变成 class，才能点出

e.g int → Integer

来 hashcode

* 当前方式可能导致 diff key, same index → Conflict

key1 → index=1 conflit

key2 →

⇒ handle conflict in buckets

③ Buckets handle multiple item

List<<key, value>>

key 是 list，可能是 O(1)

O(1) 正常情况

* hash Func: hashcode(){

查找 hashmap 的 True comparing

取决于 bucket 里有多少个 <key, pair>

* index → Bucket O(1)

* Scan Buckets O(k) k is # bucket

④ 希望 hashMap.get() = O(1)

⇒ #(Bucket) size is O(1)

✳ 保证 Buckets size 是一个常数 O(1)

↳ #Buckets 尽可能多 e.g 存100个元素有100个桶且散列平均分布

↳ #(Buckets) 多 vs 少 存的元素

☆ 0.75 = $\frac{\#\langle key, val\rangle}{\#(Buckets)}$

↳ load factor (java中)

⑤ 理论上 Hashmap.get() 不能是 O(1) { ① avg O(1) [expected]
② worst case O(N)
N is size of buckets

⇒ put

void

put(key, value)

① key → index

② index → bucket[]

③ scan bucket[] 找 pair

{ avg. O(1)
worst case O(N)

④ Insert or update

⇒ Remove remove(key)

return boolean ① key → Index

// True key exist, ② Index → Bucket[]

other False ③ Scan Bucket[] 找pair<key, value>

④ delet

⇒ boolean contained(key)

// True key exist, other False

2. 创建和初始化

a) Map<Integer, Integer> map = New HashMap<>(); → Map的subset

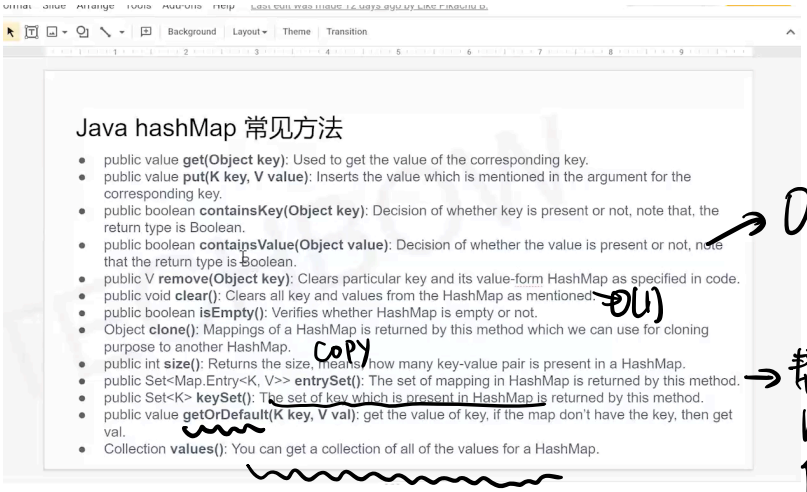
↑ Interface

..... frequenceMap = new HashMap<>();

3. 放列表

a) data 排步 not 连续，是分散排 & bucket 可能为空

4. method



Java hashMap 常见方法

- public value **get(Object key)**: Used to get the value of the corresponding key.
- public value **put(K key, V value)**: Inserts the value which is mentioned in the argument for the corresponding key.
- public boolean **containsKey(Object key)**: Decision of whether key is present or not, note that, the return type is Boolean.
- public boolean **containsValue(Object value)**: Decision of whether the value is present or not, note that the return type is Boolean.
- public V **remove(Object key)**: Clears particular key and its value-form HashMap as specified in code.
- public void **clear()**: Clears all key and values from the HashMap as mentioned.
- public boolean **isEmpty()**: Verifies whether HashMap is empty or not.
- Object **clone()**: Mappings of a HashMap is returned by this method which we can use for cloning purpose to another HashMap.
- public int **size()**: Returns the size, means how many key-value pair is present in a HashMap.
- public Set<Map.Entry<K, V>> **entrySet()**: The set of mapping in HashMap is returned by this method.
- public Set<K> **keySet()**: The set of key which is present in HashMap is returned by this method.
- public value **getOrDefault(K key, V val)**: get the value of key, if the map don't have the key, then get val.
- Collection **values()**: You can get a collection of all of the values for a HashMap.

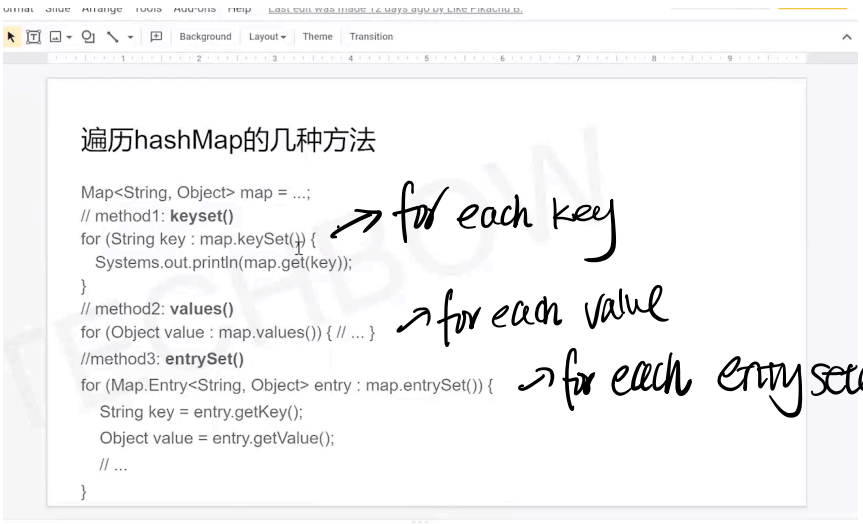
O(N)

O(1)

copy

帮你去traverse一遍 hashmap中所有的pair base on 从第0个桶到第N个桶

5. Traverse hashmap



遍历hashMap的几种方法

```
Map<String, Object> map = ...;
// method1: keyset()
for (String key : map.keySet()) {
  Systems.out.println(map.get(key));
}
// method2: values()
for (Object value : map.values()) { // ... }
//method3: entrySet()
for (Map.Entry<String, Object> entry : map.entrySet()) {
    String key = entry.getKey();
    Object value = entry.getValue();
    // ...
}
```

for each key

for each value

for each entrySet

6. 桶default开为16个 with load factor of 0.75，希望新桶是O(1)

7. Hashset

a) only key, no value

HashSet<String> hset=new HashSet<String>

key only

b) Hashmap vs hashset

HashSet vs HashMap

HashMap	HashSet
HashMap实现了Map接口	HashSet实现了Set接口
HashMap储存键值对	HashSet仅仅存储对象
使用put()方法将元素放入map中	使用add()方法将元素放入set中
HashMap中使用键对象来计算hashcode值	HashSet使用成员对象来计算hashcode值，对于两个对象来说hashcode可能相同，所以equals()方法用来判断对象的相等性，如果两个对象不同的话，那么返回false
HashMap比较快，因为是使用唯一的键来获取对象	HashSet较HashMap来说比较慢

c) e.g give student id is legal or not

e.g 结点是否被处理过

→ array 里一些 String class，不连续 int 不能表示 index