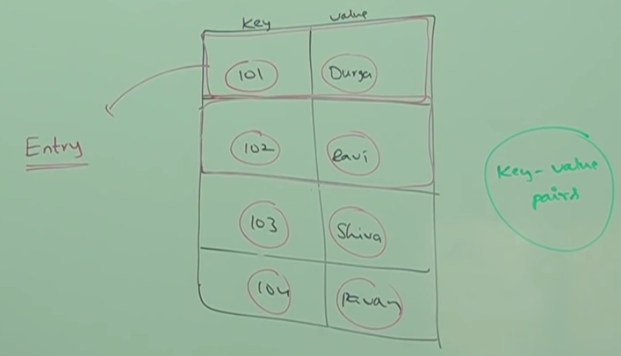


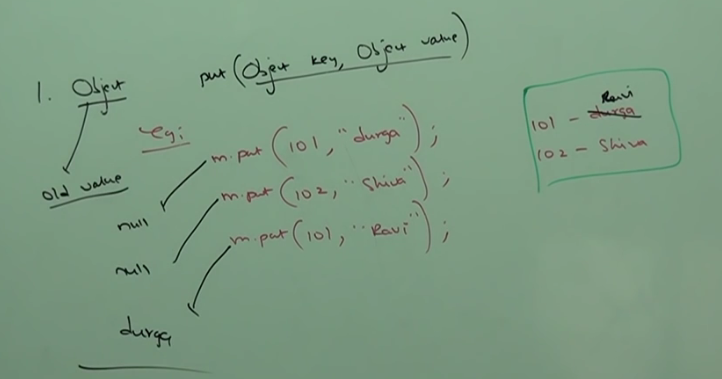
**NOTE**: Map is **not** child interface of Collection(I).  
**Application of Map**: To represent a group of objects as key, value pairs.   
**Example**:

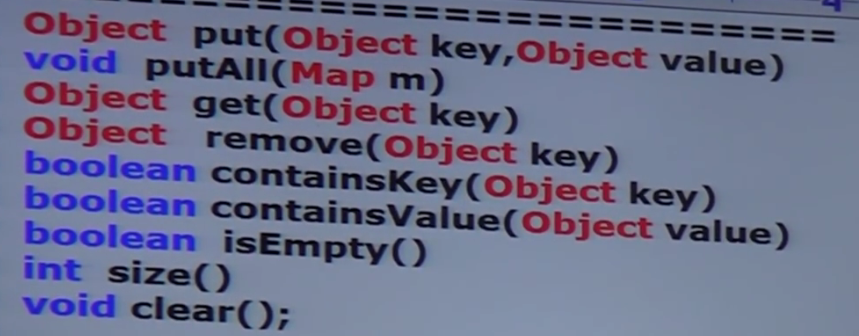


NOTE: Both key and value are objects.   
NOTE: Duplicate keys are not allowed but values.

**Entry**: Each key, value pair is called entry hence is considered as **a collection of entry objects**.

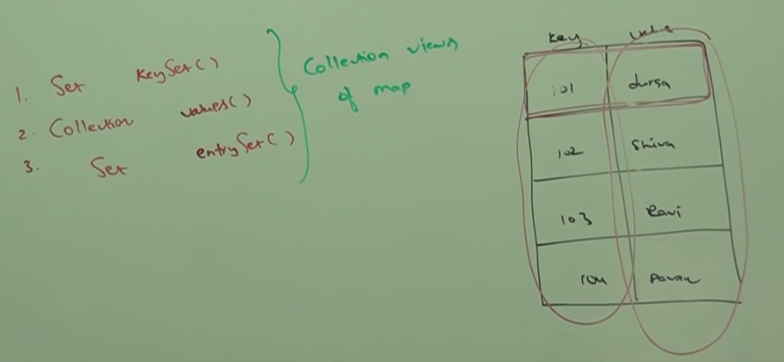
**Map Interface Methods**

1. Object put(Object key, Object value)
   1. **return**: null or the previous value replaced by the new value.   
      
2. putAll(Map m)
3. get(key):  
   null or value
4. remove(key)  
   Complete entry will be removed.
5. containsKey(key)
6. containsValue(value)
7. isEmpty()
8. size()

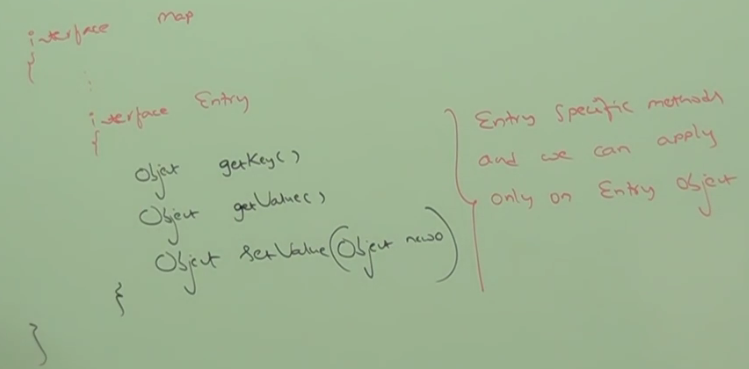


# Collection Views of map

* Set keySet()
* Collection values()
* Set entrySet()



# Entry

Map is a group of Key, Value pair and each key, value pair is called **entry**. Hence, Map is considered as a collection of Entry objects. Without existing Map object, no chance for existence for Entry objects. Hence, Entry (I) is defined inside Map (I) 

HashMap

# Characteristics of HashMap

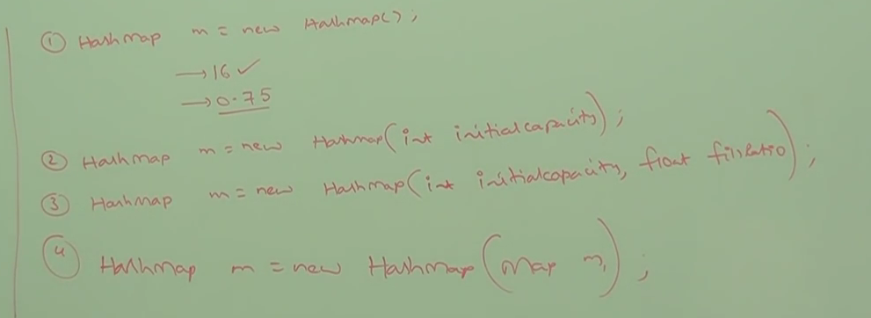
1. Underlying DataStructure is Hashtable.
2. Insertion order not preserved. Insertion is based on hashcode of keys.
3. Duplicate keys are not allowed but values.
4. Heterogeneous objects are allowed for both keys and values.
5. Null is allowed for key only once but for values for any number of times.
6. HashMap implements Serializable(I), Cloneable(I) interfaces but not ~~RandomAccess(I)~~
7. HashMap is the best choice if our frequent operation is search operation.

# Constructors

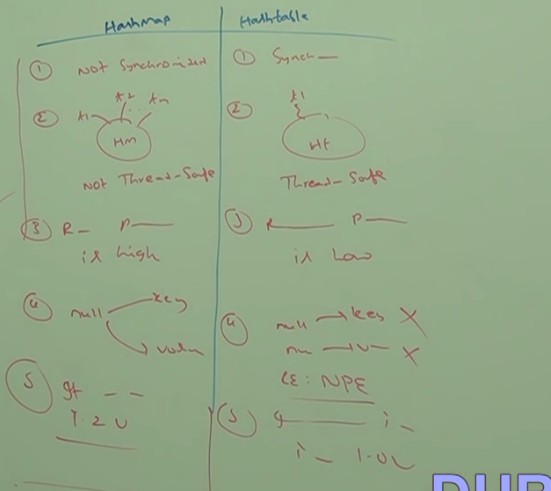
NOTE: Where there is Hash, the constructors are same. 4 Constructors.

1. HashMap()  
    Initial Capacity 🡪 16

Fill Ratio 🡪 .75 = 1/3

1. HashMap(int initialCapacity)
2. HashMap(int intialCapacity, float fillRatio)
3. HashMap(Map map)  
   
4. **NOTE**: The modification operations performed on Entry object will be reflected on the actual map object. ☺

Difference b/w HashMap and Hashtable



How to get synchronized map from non-synchronized map

By Default, HashMap is non-synchronized but we can get synchronized version of HashMap by using Collections.synchronizedMap(map) .

How to work with Map.Entry

