HashMap

* Extends AbstractMap
* Has an array of nodes ( Node has K key ,V value and int hash ,Node next )
* Hashing – process of converting an object into integer form by using the Hashcode method.
* Bucket is an element of HashMap array to store the node
* Single bucket can have more than one node (depending on Hashcode)
* Buckets are different in capacities.
* Capacity = number of Bucket \* capacity
* Index = Hashcode (key) & (N-1)
* Allows null key
* Initial size =16
* Load factor =0.75 (75 % of capacity)
* Threshold = current capacity \* load factor
* The number of entries in HashTable that exceeds the Map is rehashed, twice the number of buckets as before.
* Rehashing 🡪 recalculating the Hashcode value of already stored entries.
* Collision occurs when the hash function returns the same bucket location with 2 different keys.
* While Collision it checks for *Hashcode() and equals()* method that both the key are same.
  + If keys are the same, replace the value with the current value.
  + Otherwise, connect the node object to the previous node object through a linked list so both are stored at the same index.
* Hashmap initially uses the linkedList
  + When the entries crosses certain threshold , it will replace the linkedlist with balanced binary tree(red-black Tree)

Methods :

*Get*

*Put*

*Getor Default*

*Remove*

*Containskey*

*containsValue*

*isEmpty*

*size*

complexity

O(1) for insertion and lookup