HashMap stores elements using Hashing technique which assigns a unique hashcode to an object by applying formula to the properties of the object.

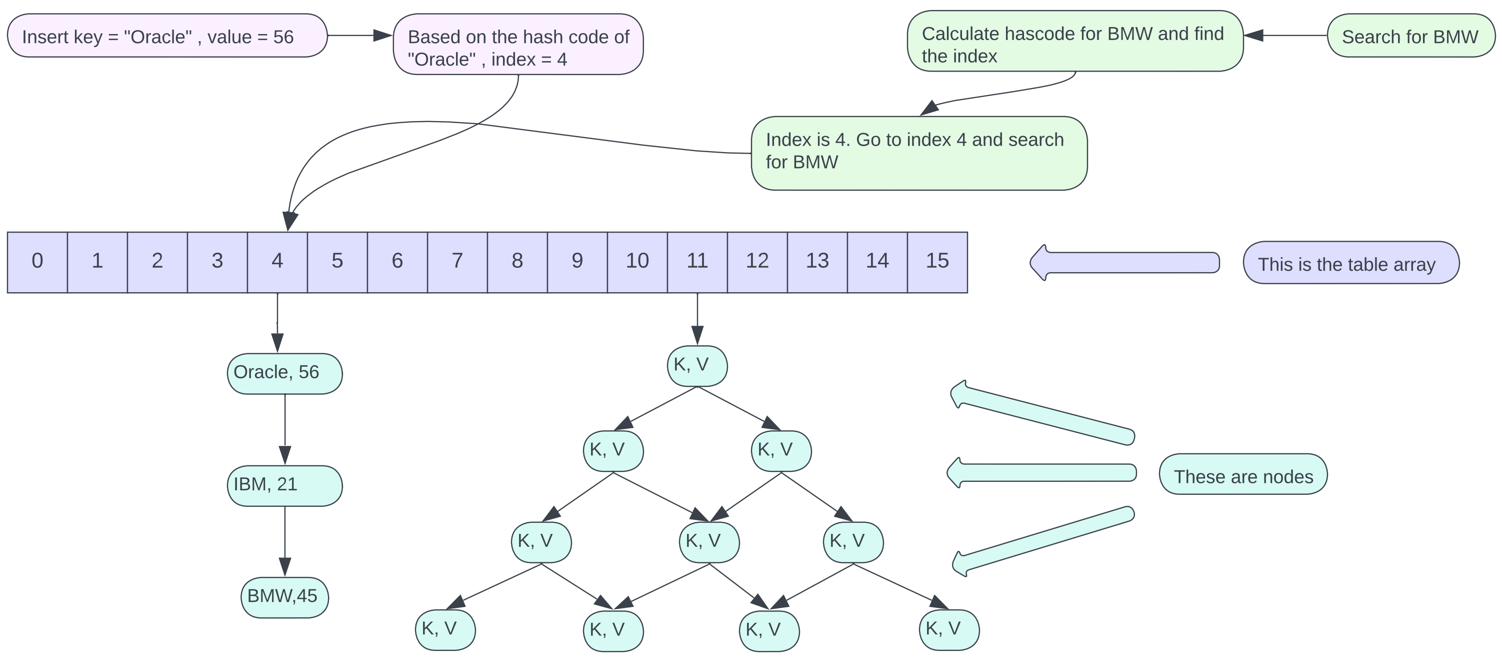
HashCode properties :
2 equal objects must have same hashcode but 2 objects having same hashcode are not necessarily equal.

HashMap has a nested static class called **Node**

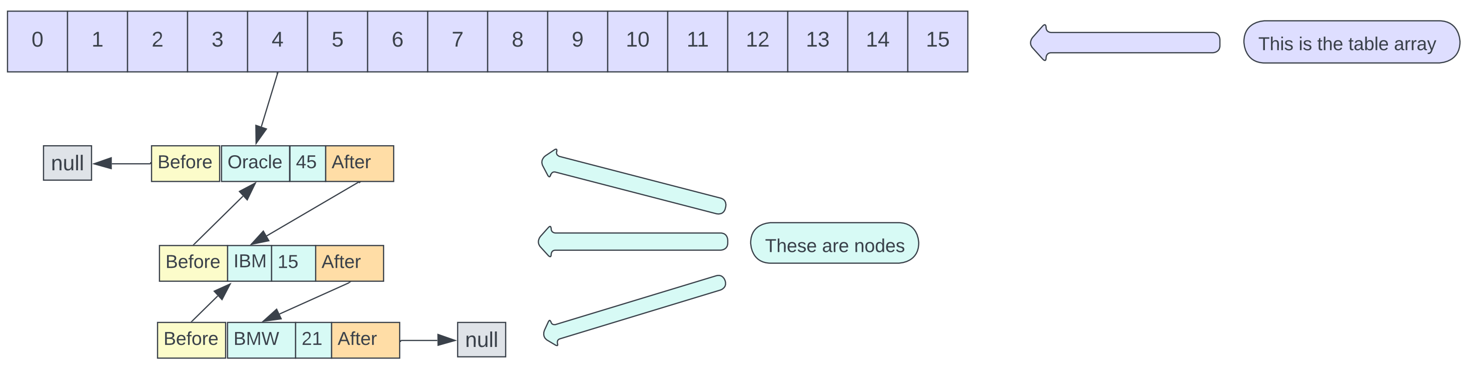
```
static class Node<K,V> implements Map.Entry<K,V> {
        final int hash;
        final K key;
        V value;
        Node<K,V> next; -- points to next node
 ..}
```

It also has a field called **table** which is an array of uninitialized Node objects. - transient Node<K,V>[] table;

- When we create a hashmap with default constructor, .75 gets assigned to DEFAULT_LOAD_FACTOR but table array is not initialized.
- When an element is inserted, the table array is initialized with size of 16, so we have 16 buckets with index of 0 to 15.
- If key is null, then it is inserted at index 0 because hashcode of null is 0.
- Else hashcode of key is calculated and based on hashcode of key, index is decided. If there is no element at that index, a new node is created and inserted at that index.
- Now, if we insert another key having same hashcode, collision occurs. In that case if key is equal, value of key is updated, else key-value pair is added at the end of exisiting key to form a linked list.
- In Java 8, an improvement was introduced. If the size of linked list becomes greater than TREEIFY_THRESHOLD which has a default value of 8, then it is converted to a red black tree.
- 6.   To get value from hashmap, hashcode of key is calculated and index is found out. There can be 0 or more keys in an index. So we match our key with all the keys in that index using equals() method and return value.


Insert key = "Oracle" , value = 56
Based on the hash code of "Oracle" , index = 4
Calculate hascode for BMW and find the index
Search for BMW
Index is 4. Go to index 4 and search for BMW
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
This is the table array
Oracle, 56
IBM, 21
BMW,45
K, V
These are nodes

This is a **linkedHashMap** where the key-values are stored in a doubly linked list. A doubly linked list has node which keeps record of both next and previous and can track insertion order.


0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
This is the table array
null
Before | Oracle | 45 | After
Before | IBM | 15 | After
Before | BMW | 21 | After
null
These are nodes