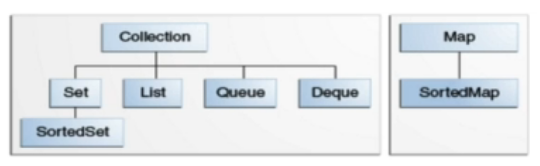
**Collections**



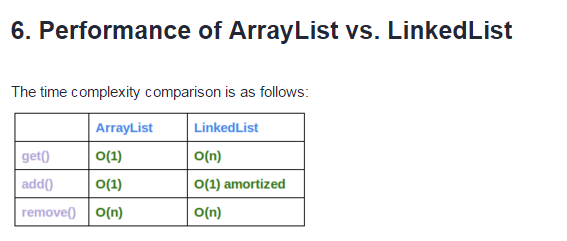
**Array list – (or) Dynamic Arrays**

Any elements added in middle (or) Anything to be deleted 🡪

array list has to be shifted accordingly

**Linked List** – Subsequent address is used to add the elements. Any addition it just relinks. This need to shift anything down. Due to this reason it is pretty fast. In case an element is removed – Java automatically performs garbage collection

Java has implemented doubly linked list. Hence it is easy to traverse back

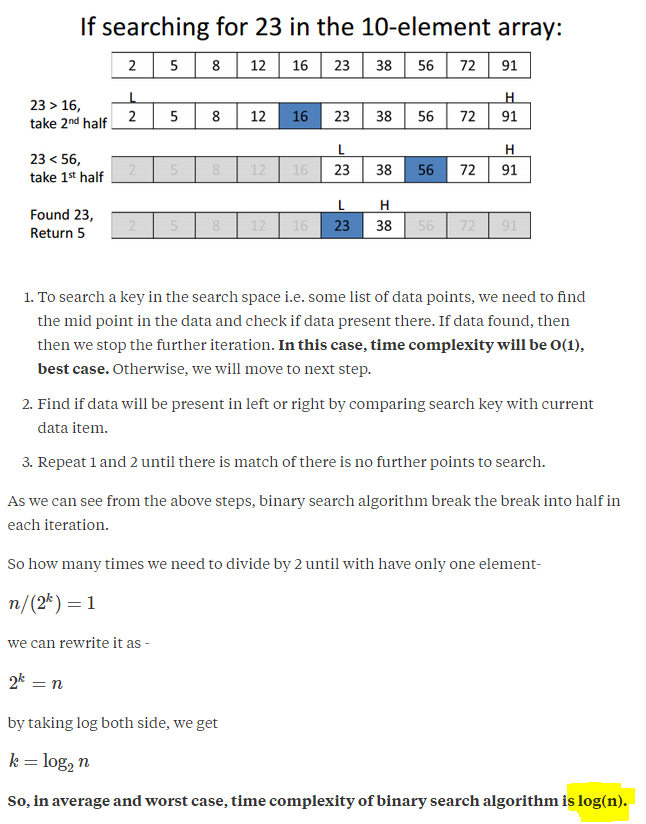


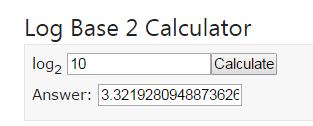
Similarly when you insert then it becomes

order(n) order(1) in case of Linked List

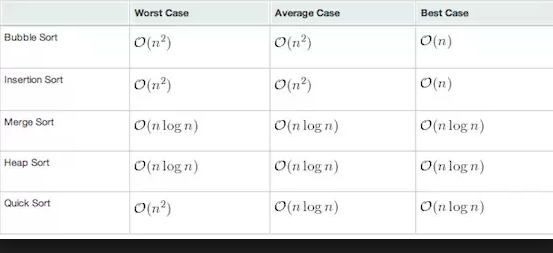
**Binary Search**

If the List is sorted array then binary search is very quick in searching the elements. The idea of binary search is divided the elements and search is performed.





**Sorting Techniques**



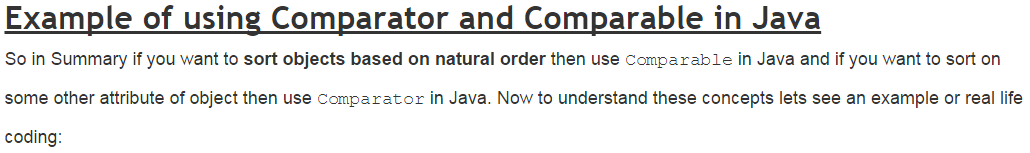
In Java, Collections.sort() method uses merge sort

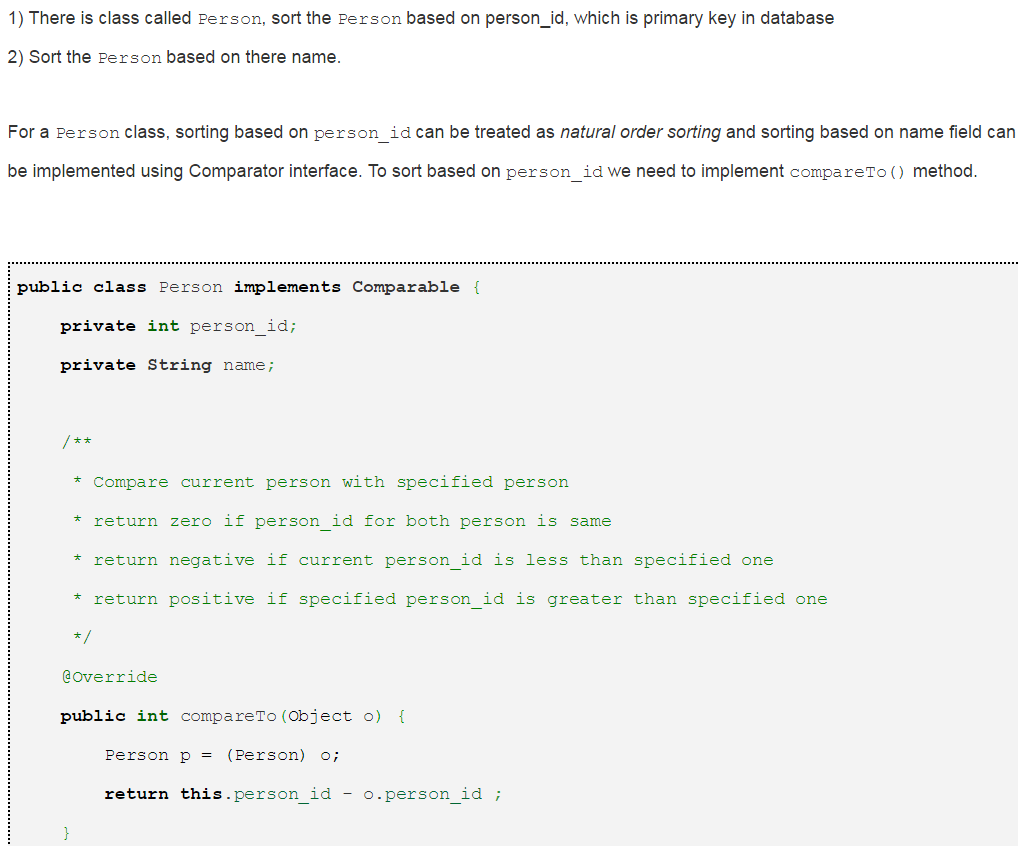
**Sorting Techniques**

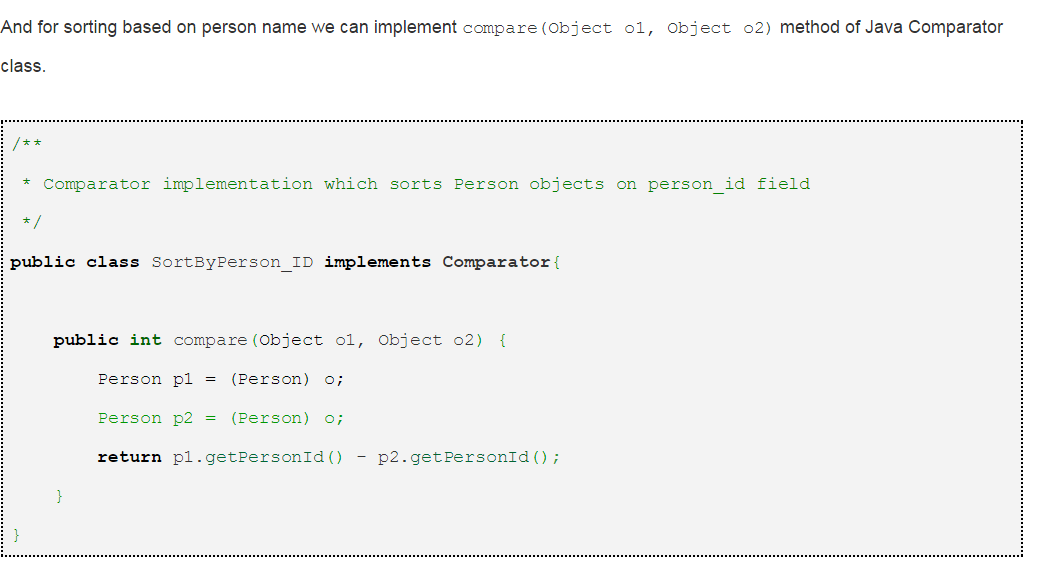
* Comparable and Comparator both are sorting techniques which can be over ridden and performed

Comparable – Typically comparing its own instance with another object

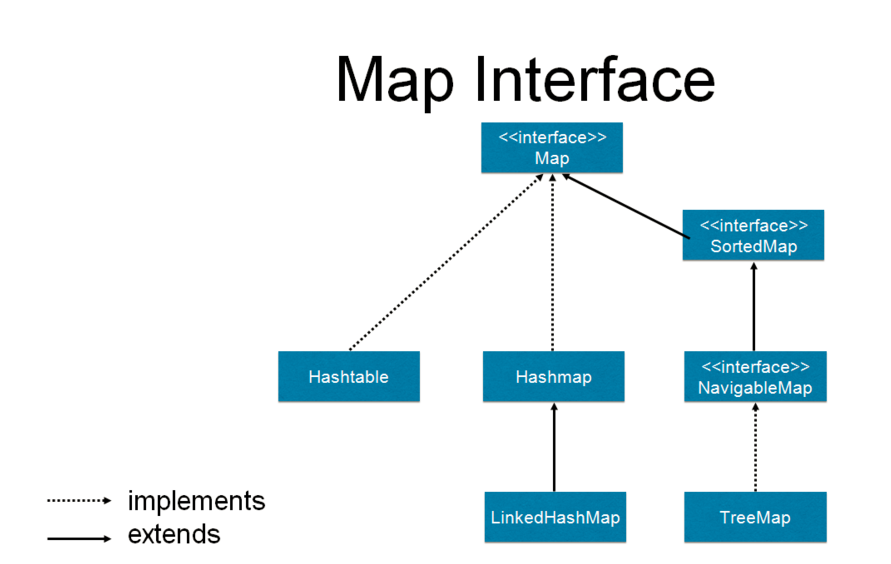
Comparator – Comparing two different instances.



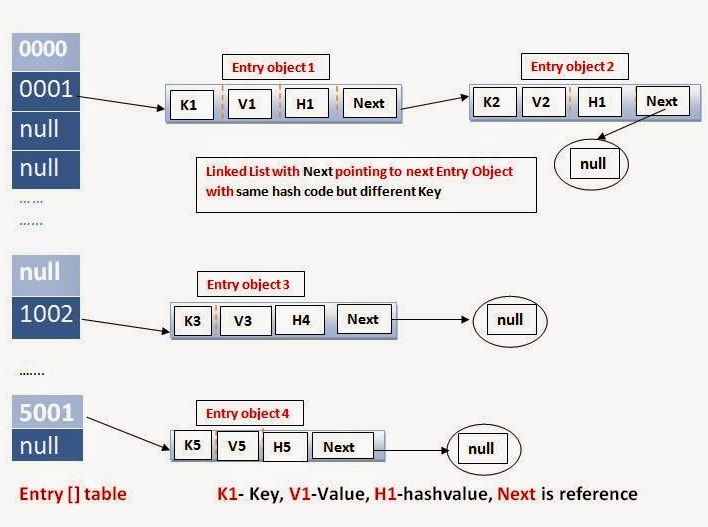




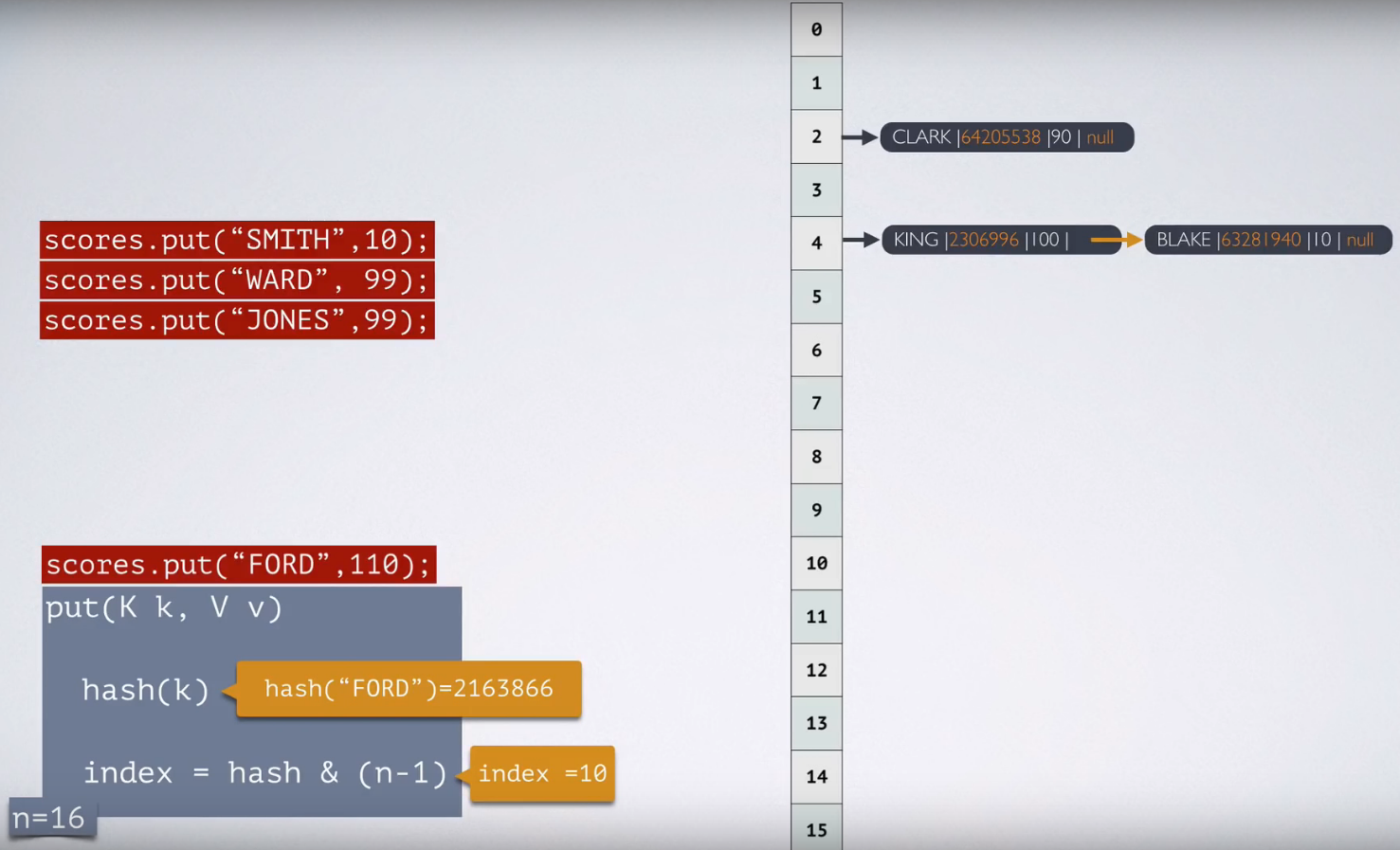
**Maps –** Map is not under collection interface put as part of Collections util package



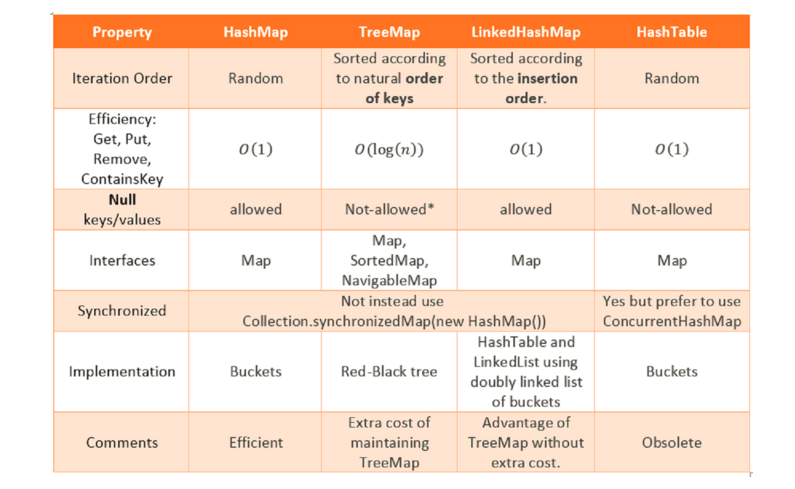
* Internally Hash map is a collection of tables(Buckets) with a Linked List



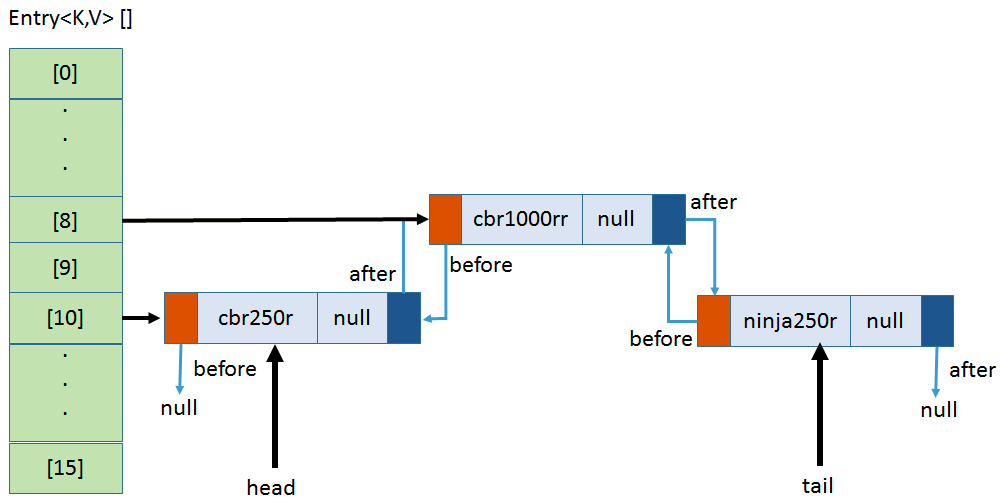
Hashed Keys which are nothing but Hash codes are separated into different buckets. Whenever there is a collision in the bucket, Hash Key values pair are arranged accordingly.



**Difference between variations of Maps collections**



**Linked Hash map** maintains the order of insertion keys using doubly linked list. From the below you can notice each key value pair is piggy backed using doubly linked list



**Tree map**