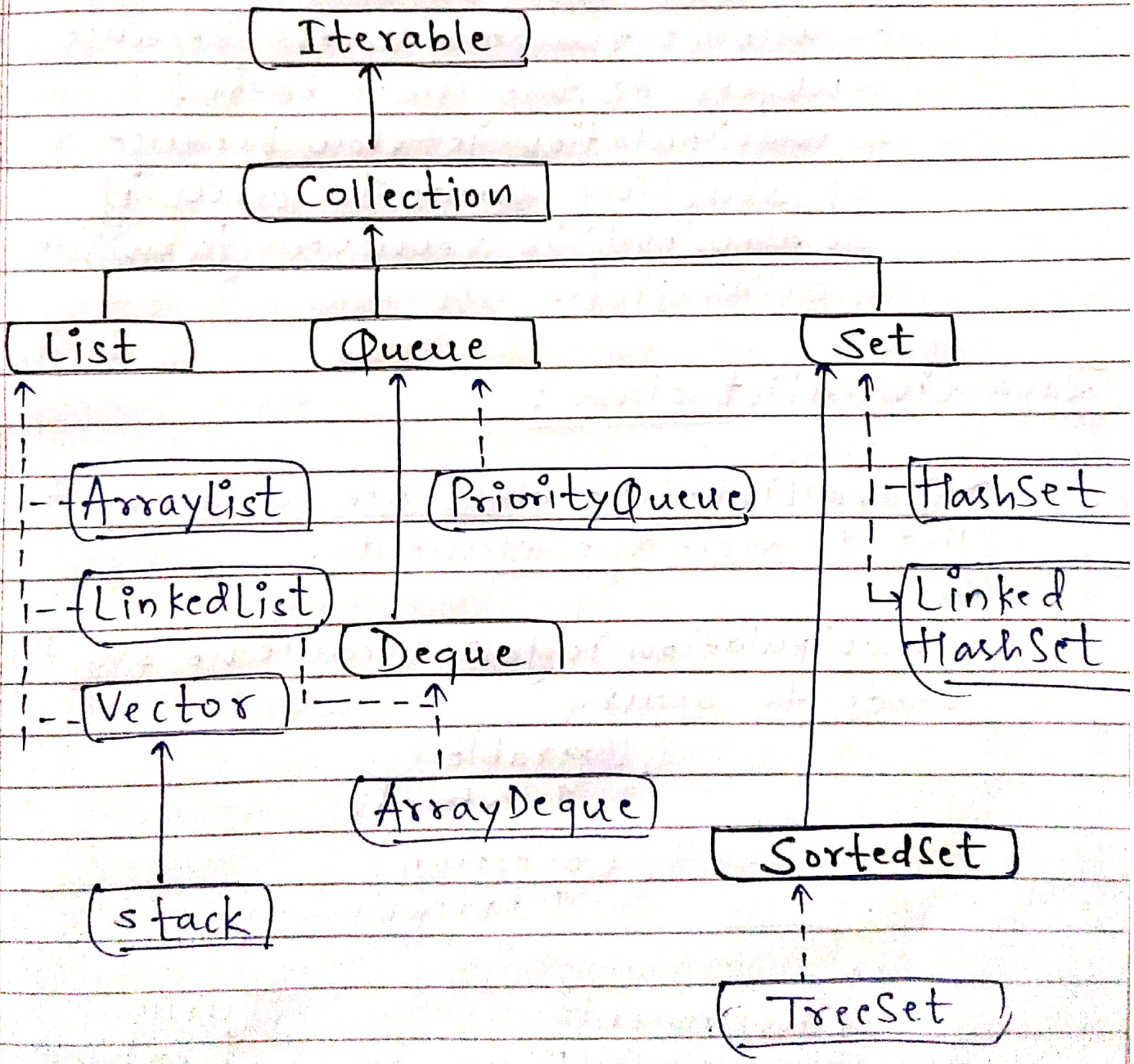
Collections in Java :

A collection represents a single unit of objects, i.e a group.
The Collection Framework represents a unified architecture for storing and manipulating a group of objects.

Hierarchy



Scanned by CamScanner

Java Arraylist class :

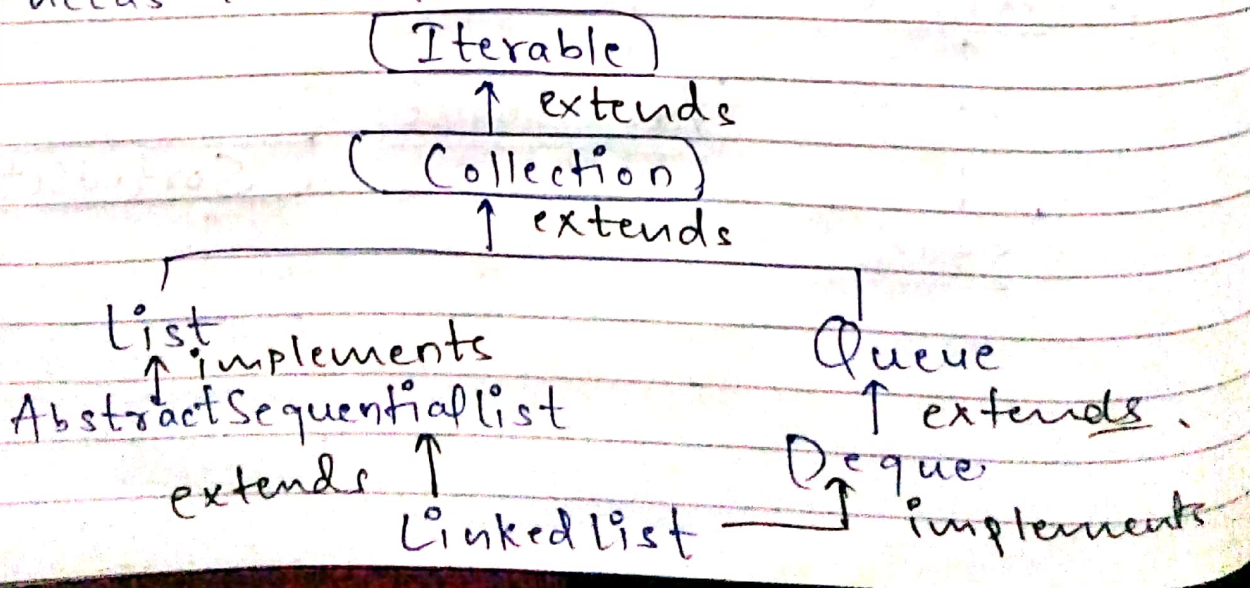
> Java Arraylist class uses a dynamic array for storing the elements. It inherits Abstractlist class and implements List interface.

> Important points
- can contain duplicate elements
- maintains insertion order
- is non-synchronized
- allows random access because array works at the index basis.
- manipulation is slow because a lot of shifting needs to occur if any element is removed from the array list.

Java LinkedList class :

> Java LinkedList class uses a doubly linked list to store the elements.

> manipulation is fast because no shifting needs to occur.



Scanned by CamScanner

Difference Between Arraylist & Linked list

Arraylist	Linked list
> Internally uses a dynamic array to store the elements.	> Internally uses a doubly linked list to store elements.
> Manipulation is slow because it internally uses an array. If any element is removed from the array, all the bits are shifted in the memory.	> Manipulation is faster because it uses a doubly Linked list, so no bit shifting is required in memory.
> can act as a list only because it implements list only.	> can act as list and queue because it implements List and Deque interfaces
> Better for storing and accessing data.	> better for manipulating data.

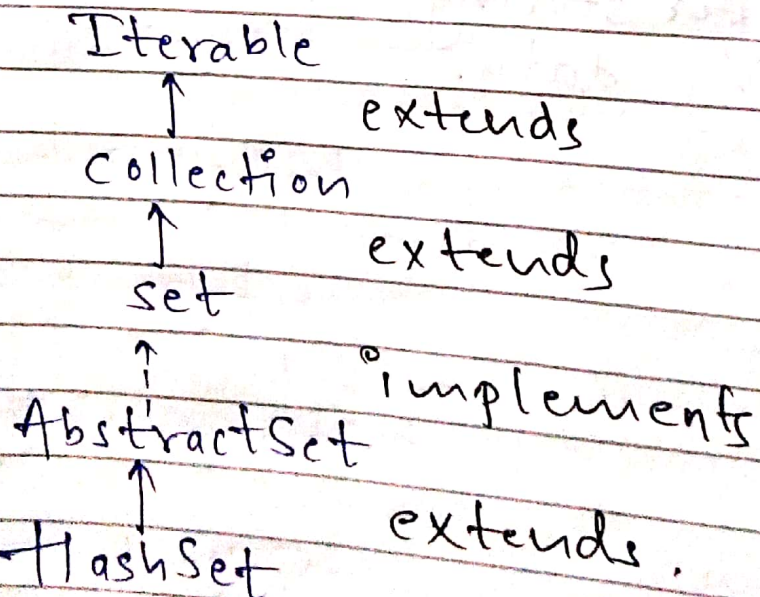
Scanned by CamScanner

Java HashSet :

> class is used to create a collection that uses a ~~hab~~ hash table for storage.

> Important points:
- stores element by using a mechanism called hashing
- contains unique elements only.
- allows null value
- class is non synchronized
- doesn't maintain order of insertion, here elements are inserted on the basis of their hashcode.
- Best approach for search operations.
- initial default capacity of Hashset is 16; and the load factor is 0.75.

Heirarchy



Scanned by CamScanner

Working :

It constructs a collection that uses a hash table for storing elements. It contains unique elements.
It inherits the Abstractset class & implement the Set interface.
It uses a technique to store elements is called hashing.

Hashset uses HashMap internally in Java

> Hashset implements Set interface. It guarantee uniqueness. It is achieved by storing elements as keys with the same value always.

> Hashset ~~always~~ does not have any method to retrieve the object from Hashset. There is only way to get objects from Hashset via Iterator.

> When we create an object of Hashset, it internally creates an instance of HashMap with default initial capacity 16.

HashSet uses a constructor Hashset (int capacity) that represents how many elements can be stored in the Hashset. The capacity may increase automatically when more elements to be store.

Scanned by CamScanner

Java LinkedHashSet class :

is a Hashtable and LinkedList implement of the set interface. It inherits HashSet class and implements Set interface.

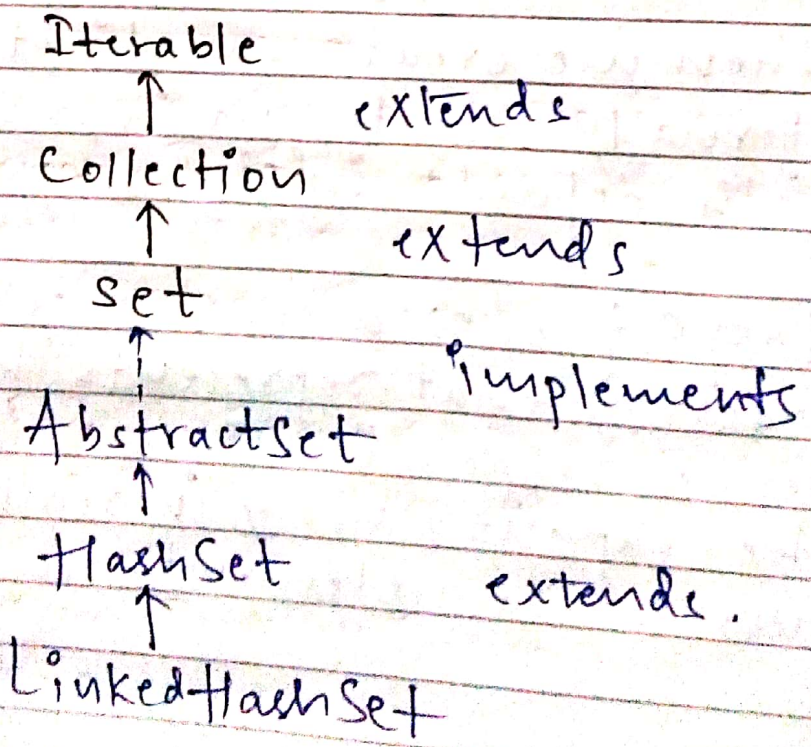
Points :

> contains unique elements only like HashSet
> provides all optional set operation and permits null elements.
> is non-synchronized.
> maintains insertion Order.

declaration –

public class LinkedHashSet<E> extends HashSet implements Set<E>, Cloneable, Serializable

Heirarchy –



Scanned by CamScanner

Java Treeset class:

> Implements the Set interface that uses a tree for storage. It inherits AbstractSet class and implements the NavigableSet interface.

> The objects of the Treeset class are stored in ascending order.

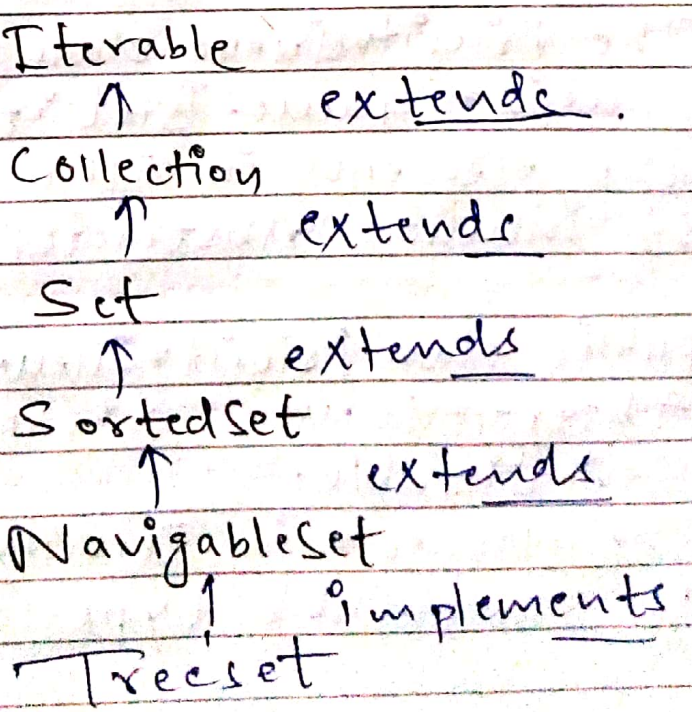
Points –

> contains unique elements like HashSet.
> access and retrieval times are quiet fast.
> class doesn't allow null elements
> non synchronized.
> maintains ascending order.

declaration:

public class TreeSet<E> extends AbstractSet<E> implements NavigableSet<E> cloneable, Serializable

Hierarchy:



Scanned by CamScanner

Queue And PriorityQueue

Java Queue Interface –

> orders the element in FIFO manner. In FIFO first element is removed first & last element is removed at last.

public interface Queue<E> extends Collection

Methods of Queue Interface –

> boolean add(object) – // returns true upon success
> boolean offer(object)
> Object remove()
> Object poll() – // returns null if q emp
> Object element()
> Object peek() – returns null if this queue is empty.

PriorityQueue Class :

The PriorityQueue class provides the facility of using queue. But it does not orders the elements in FIFO manner.
It inherits AbstractQueue class.

public class PriorityQueue<E> extends AbstractQueue<E> implement Serializable.
The elements in PriorityQueue must be of comparable type.

Scanned by CamScanner

Java Deque Interface :

is a linear collection that supports element insertion and removal at both ends.
Deque is an acronym for "double ended queue".

public interface Deque<E> extends Queue<E>

ArrayDeque class :

provides the facility of using deque and resizable-array.
It inherits AbstractCollection class and implements the Deque interface.

- > we can add or remove elements from both sides
- > Null elements are not allowed in ArrayDeque.
- > is not threadsafe, in presence of external synchronization.
- > no capacity restrictions.
- > is faster than LinkedList & Stack.

Scanned by CamScanner

Java HashTable class :

- implements a hashtable, which maps keys to values. It inherits Dictionary class and implements the Map Interface.

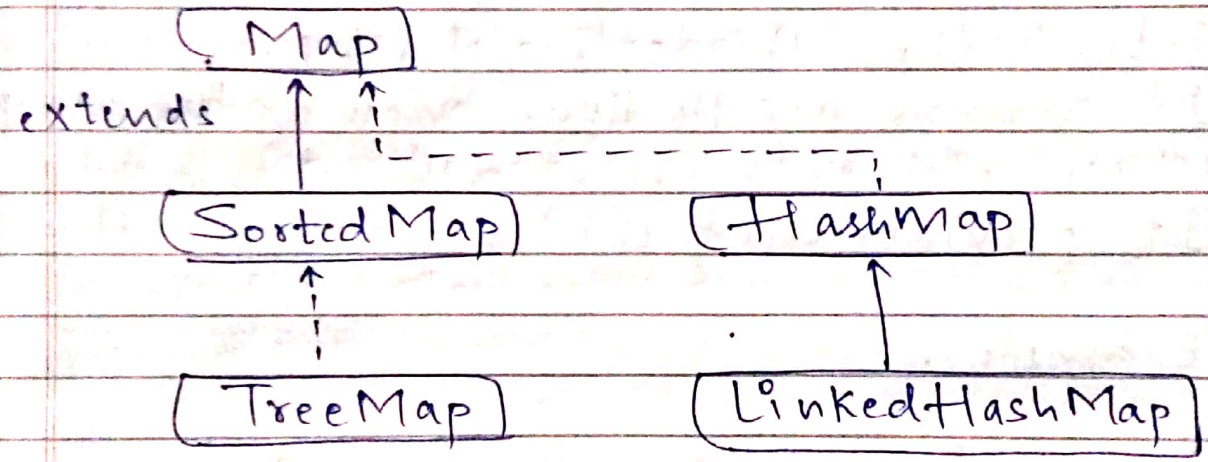
- Points
  - A HashTable is an array of a list. Each list is known as bucket. The position of bucket is identified by calling the hashcode() method.
  - A HashTable contains values based on keys.
  - doesn't allow null key or value.
  - class is synchronized.
  - default capacity is 11 whereas loadfactor is 0.75.

Scanned by CamScanner

Java Map Interface :

- A map contains values on the basic of key, i.e key and value pair.
- Each key and value pair is known as an entry
- A Map contains unique keys.
- A Map is useful if you have to search, update or delete element on the basis of a key.

Hierarchy :



- A map doesn't allow duplicate keys, but you can have duplicate values.
- HashMap and LinkedHashMap can have i.e allow null keys and values but TreeMap doesnt allow any null key or value.
- A Map can't be traversed, so you need to convert it into Set using keyset() or entrySet() method.

Scanned by CamScanner

HashMap - is the implementation of Map, but it doesn't maintain any order.

LinkedHashMap - is the implementation of Map.
> It inherits HashMap class
> It maintains insertion Order.

TreeMap - is the implementation of Map and SortedMap.
> It maintains ascending order.

Map.Entry Interface :

> Entry is the subinterface of Map.
> It returns a collection-view of the map, whose elements are of this class.
> It provides methods to get key and value.

Example :

```
import java.util.*;

class MapExample
{
 public static void main(String args[])
 {
    Map<Integer, String> map =
                  new HashMap<Integer, String>();

    map.put(1, "Abhi");
    map.put(2, "Avi");
    map.put(3, "Axu");
```

Scanned by CamScanner

```
Set set = map.entrySet();
Iterator itr = set.iterator();

while(itr.hasNext())
{
Map.Entry entry = (Map.Entry) itr.next();

sop(entry.getKey()+" "+entry.getValue());

}

}

}
```

JavaHashMap class :

> implements the map interface by using a hash table.

> Points-
- contains values based on the key.
- contains only unique keys
- may have one null key and multiple null values.
- non-synchronized.
- maintains no order.
- initial default capacity of JavaHashMap class is 16 with a load factor 0.75.

Scanned by CamScanner

mycompanion

public class HashMap<K,V> extends
AbstractMap<K,V> implements
Map<K,V>, cloneable, Serializab

K: It is the type of keys maintained by this map.
V: It is the type of mapped values.

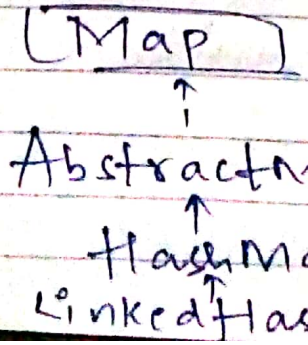
Hashset contains only values whereas HashMap contains an entry (key and value).

Java LinkedHashMap class:

Java LinkedHashMap class is Hashtable and linked list implementation of the Map interface, with predictable iteration order.

Points -

- > contains values based on the key
- > contains unique elements
- > may have one null key & multiple null values.
- > non-synchronized
- > maintains insertion order
- > capacity is 16 with a load factor of 0.75.



Scanned by CamScanner

Java TreeMap class :

> Java TreeMap class is a red-black tree based implementation.

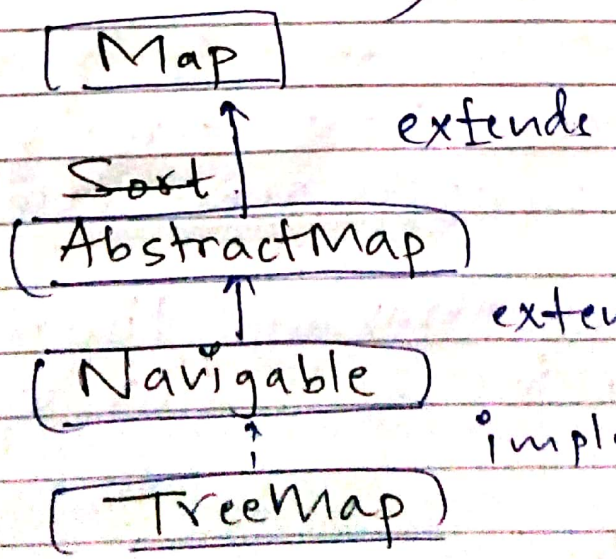
> It provides an efficient means of storing key-value pairs in sorted order.

> Points

- contains values based on key.
It implements NavigableMap interface and extends AbstractMap class.

- contains only unique elements.
- cannot have a null key but can have multiple null values.
- non synchronized
- maintains ascending order.

public class TreeMap<K,V> extends AbstractMap <K,V> implements NavigableMap<K,V>, Cloneable, Serializable



Scanned by CamScanner

HashMap	TreeMap
> can contain one null key	> cannot contain any null key
> maintains no order	> maintains ascending order.

Working of HashMap in Java :-

Hashing

It is the process of converting an object into an integer value.

The integer value helps in indexing and faster searches.

HashMap

HashMap uses technique called Hashing.
It implements map interface.
It stores the data in the pair of key and value.

> HashMap contains an array of the nodes, and the node is represented as a class.

It uses an array and LinkedList data structure internally for storing key and value.

Scanned by CamScanner

Node<K,V>
int hash
K key
V value
Node <K,V> next

> equals() :

- It checks the equality of two objects. It compares key, whether they are equal or not.
- It is a method of Object class.
- It can be overridden.

> If you override the equals() method, then it is mandatory() to override the hashCode() method.

> hashCode()

- method of the Object class.
- returns the memory reference of the object in integer form.

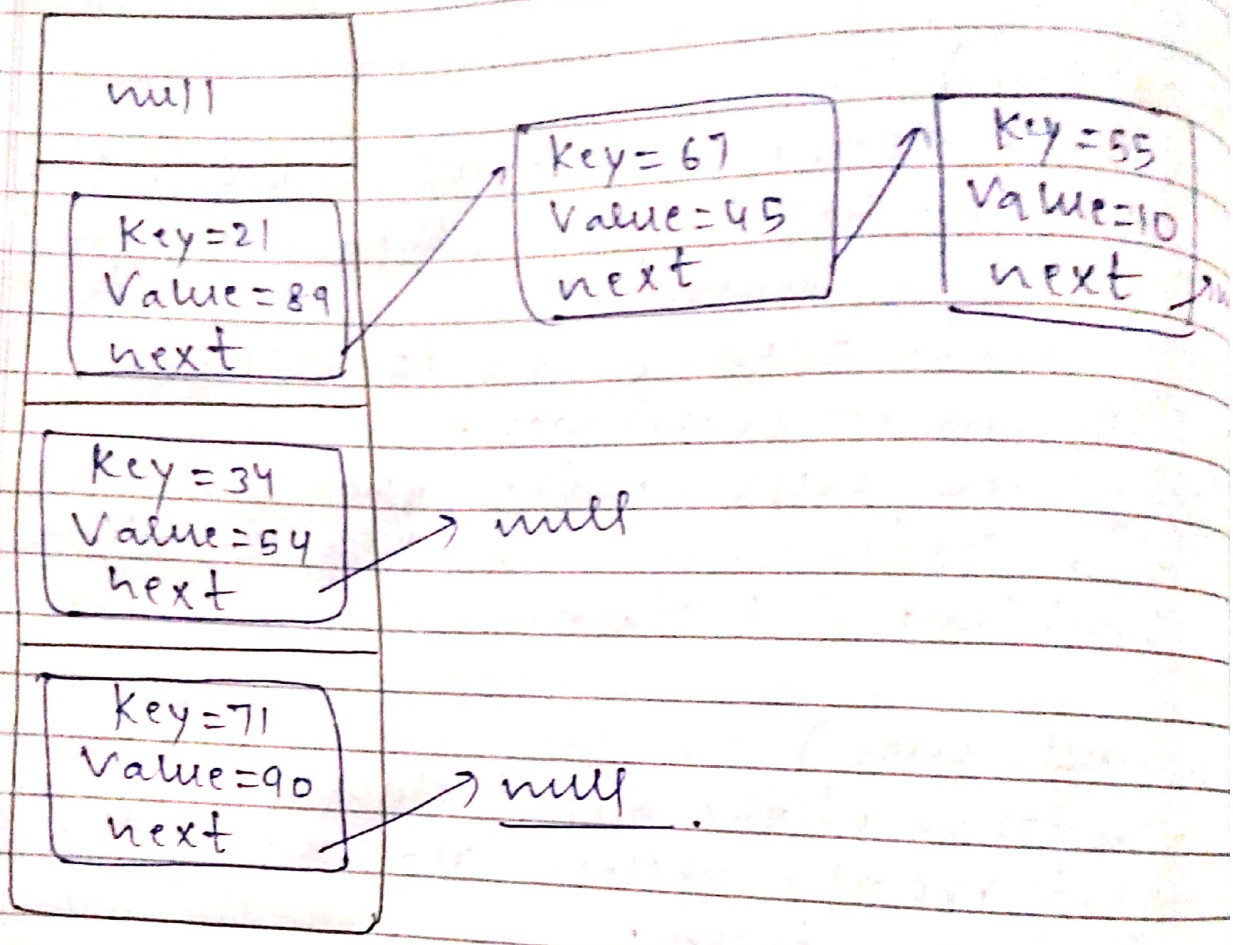
- The value received from the method is used as bucket number. The bucket number is the address of the element inside the map.
Hashcode of null key is 0.

> Buckets :

- Array of the node is called buckets.
- Each node has a data structure like a Linked list.

Scanned by CamScanner

More than one node can share the same bucket. It may be different in capacity.



Calculating Index:

> Index minimizes the size of the Array.

Index = hashCode(key) & (n-1);

Hash Collision:

This is the case when the calculated index value is the same for two or more keys.

Scanned by CamScanner