=> Hashtable :-

-Hashtable is a direct implemented class of Map interface which is present in java.util package

Syntax : public class Hashtable extends Dictionary implements Map, Cloneable, Serializable { - }

-Hashtable was introduced in JDK 1.0 version

-Hashtable is also known as legacy class

-The underline data structure of Hashtable is "Hashtable"

Properties of Hashtable :-

1. Hashtable stores the values in key-value pair and each key-value pair is known as entry

2. In Hashtable, keys should always unique but values can be duplicate

3. Hashtable can store heterogeneous elements at key position

4. In Hashtable we cannot insert null values at key or value position

5. Hashtable does not follows the insertion order by default

6. Hashtable does not follows the sorting order by default

7. Hashtable is synchronized map

8. Hashtable does not allows more than one thread at one time

9. Hashtable allows the sequential execution

10. Hashtable increases the execution time which in turn makes our application slow

11. Hashtable is thread-safe

Hahscode Method Demo:

```
public class Test {
    public static void main(String[] args) {
        Test t1 = new Test() ;
        System.out.println(t1.hashCode());

        Test t2 = new Test() ;
        System.out.println(t2.hashCode());

        Integer i = 101 ;
        System.out.println(i.hashCode());  //for integer hascode value is
same as that of value

        String str1 = new String("Arun") ;
        System.out.println(str1.hashCode());

        String str2 = new String("Arun") ;
        System.out.println(str2.hashCode());
    }
```

Output:

```
2065951873
1791741888
101
2049706
2049706
```

When we should use Hashtable :-

= Hashtable is good for searching or retrieval operation

=> Working of Hashtable :-

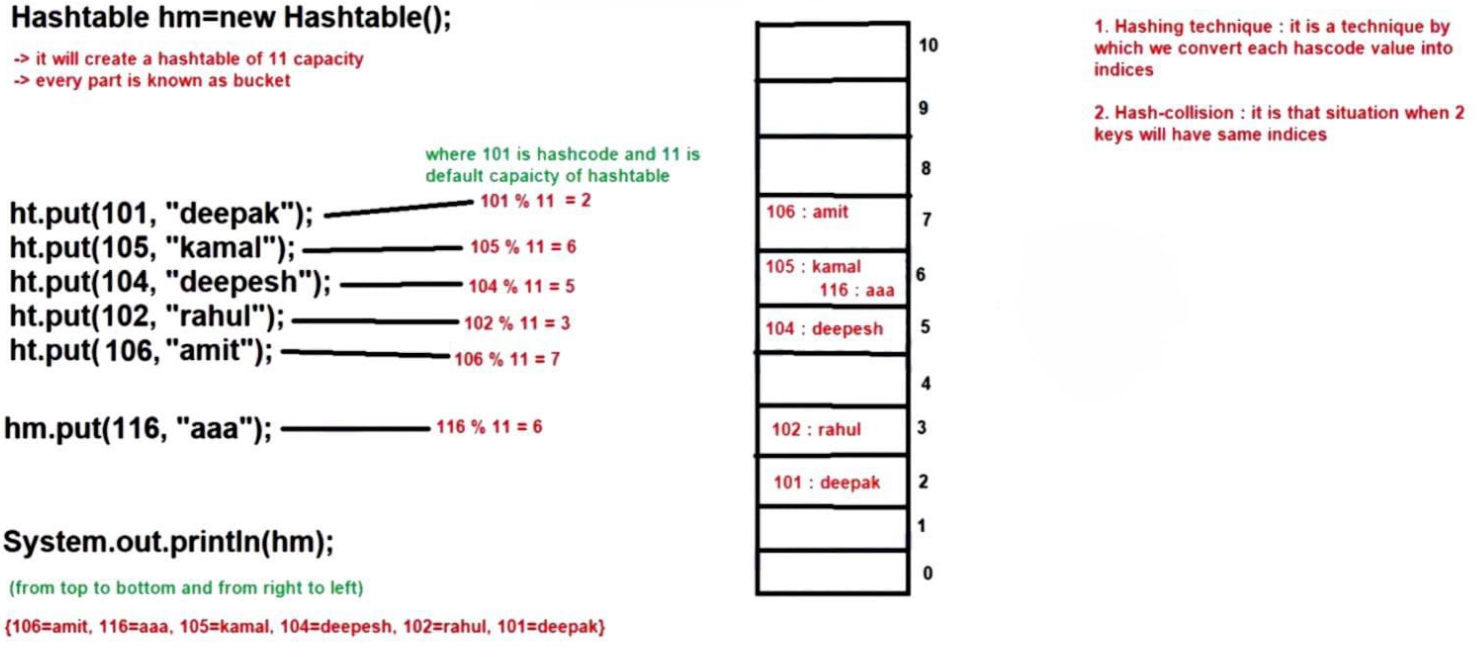
-"hashcode" is the unique integer value of each and every object that is provided by JVM

-Hashtable initialCapacity is 11

-Then for each and every key hashcode value will be generated and its index position will be calculated by using hashing technique And at that index position that key-value pair or entry will be inserted

-If two elements have same index position, then that entry will be inserted at right side of previous entry

-When the values are traversed then they are traversed from top to bottom and right to left



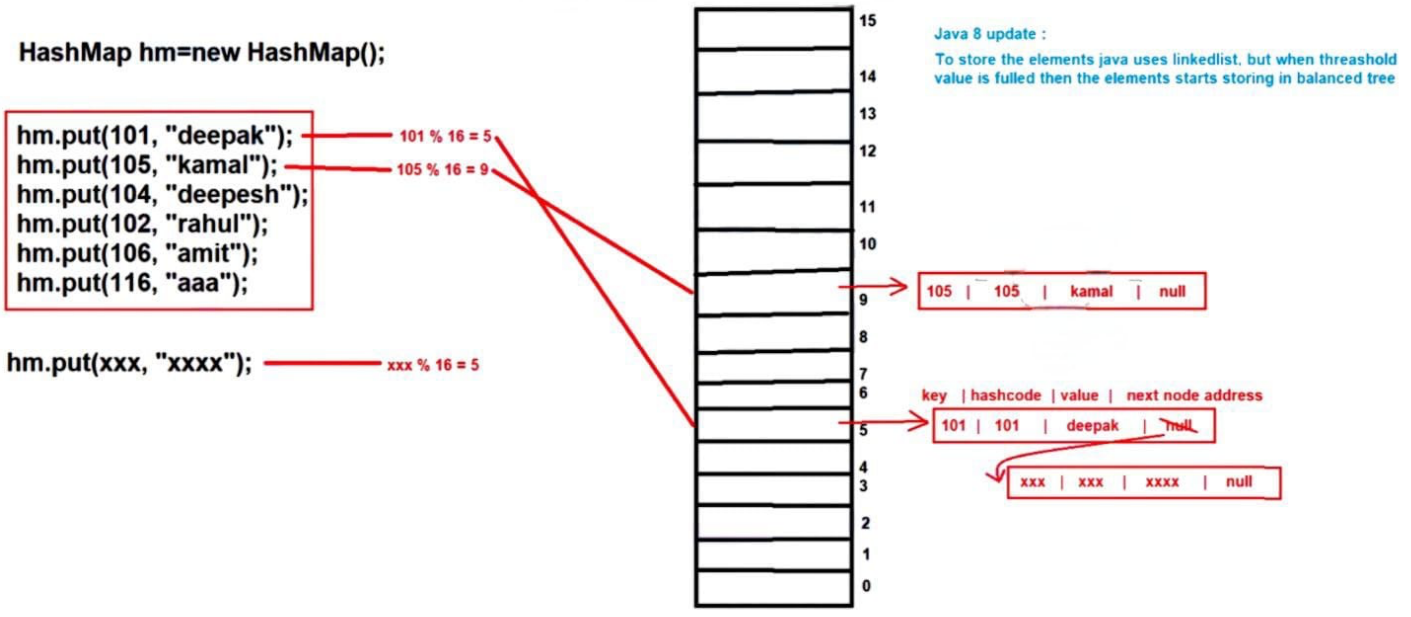
=> Working of HashMap :-

-HashMap initialCapacity is 16

-For every entry it will calculate the index position and store the element there.

-If multiple entries have same index position, then it will create linked list and starts storing in that linked-list

-In java 8 updates, after threshold value is filled then it starts storing the elements in balanced tree



Can we override the hashcode() method --> Yes

When we want to provide hashcode value of our own

```
class A
{
    int i;
    A(int i)
    {
        this.i=i;
    }
    @Override
    public int hashCode()
    {
        return i;
    }
    @Override
    public String toString()
    {
        return i+"";
    }
}
public class Test1
{
    public static void main(String[] args)
    {
        A ob1=new A(1);
        System.out.println(ob1.hashCode());

        A ob2=new A(2);
        System.out.println(ob2.hashCode());
    }
}
```

=> Difference between HashMap & Hashtable :-

1. HashMap was introduced in 1.2 version
Hashtable was introduced in 1.0 version

2. HashMap is not a legacy class
Hashtable is a legacy class

3. In HashMap we can store the null values
In Hashtable we cannot store the null value at key or value position

4. HashMap is non-synchronized Map because HashMap does not contain any synchronized methods
Hashtable is synchronized Map because it contains synchronized methods

5. Multiple points related to synchronization