**Set:** is an Interface -- is a collection that **cannot contain duplicate elements. Does not maintain insertion order. Allows single null values.**

**Set:**

There are **three main implementations** of **Set Interface**, **HashSet, TreeSet** and **LinkedHashSet**.

**HashSet:** implements the Set interface, backed by a hash table (actually a HashMap instance).

It makes no guarantees as to the iteration order of the set.

HashSet doesn’t maintain any order. Doesn’t allow duplicates. If we try to add duplicate the old value be overwritten. HashSet allows null values.

// HashSet declaration

HashSet<String> hset = new HashSet<String>();

// Adding elements to the HashSet

hset.add("Apple");

hset.add("Mango");

hset.add("Grapes");

hset.add("Orange");

hset.add("Fig");

//Addition of duplicate elements

hset.add("Apple");

hset.add("Mango");

//Addition of null values

hset.add(null);

hset.add(null);

//Displaying HashSet elements

System.out.println(hset);

**HashSet Methods:**

1. **boolean add(Element e)**: It adds the element e to the list.

2. **void clear():** It removes all the elements from the list.

3. **Object clone():** This method returns a shallow copy of the HashSet.

4. **boolean contains(Object o):** It checks whether the specified Object o is present in the list or not. If the object has been found it returns true else false.

5. **boolean isEmpty():** Returns true if there is no element present in the Set.

6. **int size():** It gives the number of elements of a Set.

7. **boolean(Object o):** It removes the specified Object o from the Set.

**LinkedHashSet:** maintains the insertation order. Elements get sorted in the same sequence in which they have been added to Set.

// LinkedHashSet of String Type

LinkedHashSet<String> lhset = new LinkedHashSet<String>();

// Adding elements to the LinkedHashSet

lhset.add("Z");

lhset.add("PQ");

lhset.add("N");

lhset.add("O");

lhset.add("KK");

lhset.add("FGH");

System.out.println(lhset);

// LinkedHashSet of Integer Type

LinkedHashSet<Integer> lhset2 = new LinkedHashSet<Integer>();

// Adding elements

lhset2.add(99);

lhset2.add(7);

lhset2.add(0);

lhset2.add(67);

lhset2.add(89);

lhset2.add(66);

System.out.println(lhset2);

**LinkedHashSet:**

**LinkedHashSet:** maintains insertion order, elements get sorted in the same sequence in which they have been added to the Set

LinkedHashSet implements as hash table with linked list running through it, orders its elements based on order in which they were inserted into set.

It is similar to HashSet and TreeSet, HashSet **doesn't maintain any kind of order of its elements** and TreeSet sorts elements in ascending order.

LinkedHashSet maintains the insertion order. Elements get sorted in the same sequence in which they have been added to the set.

// LinkedHashSet of String Type

LinkedHashSet<String> lhset = new LinkedHashSet<String>();

// Adding elements to the LinkedHashSet

lhset.add("Z");

lhset.add("PQ");

lhset.add("N");

lhset.add("O");

lhset.add("KK");

lhset.add("FGH");

System.out.println(lhset);

// LinkedHashSet of Integer Type

LinkedHashSet<Integer> lhset2 = new LinkedHashSet<Integer>();

// Adding elements

lhset2.add(99);

lhset2.add(7);

lhset2.add(0);

lhset2.add(67);

lhset2.add(89);

lhset2.add(66);

System.out.println(lhset2);

**TreeSet:**

**TreeSet:** sorts the elements in the ascending order. Allows null elements

TreeSet stores its elements in a red-black tree, orders its elements based on their values.

TreeSet is similar to HashSet except that it sorts the elements in ascending order while HashSet doesn't maintain any order. TreeSet allows null elements.

Like most of the other collection classes this class is also not synchronized. however it can be synchronized explicitly.

// TreeSet of String Type

TreeSet<String> tset = new TreeSet<String>();

// Adding elements to TreeSet<String>

tset.add("ABC");

tset.add("String");

tset.add("Test");

tset.add("Pen");

tset.add("Ink");

tset.add("Jack");

//Displaying TreeSet

System.out.println(tset);

// TreeSet of Integer Type

TreeSet<Integer> tset2 = new TreeSet<Integer>();

// Adding elements to TreeSet<Integer>

tset2.add(88);

tset2.add(7);

tset2.add(101);

tset2.add(0);

tset2.add(3);

tset2.add(222);

System.out.println(tset2);