**Set接口**  
Set不允许包含相同的元素，如果试图把两个相同元素加入同一个集合中，add方法返回false。  
Set判断两个对象相同不是使用==运算符，而是根据equals方法。也就是说，只要两个对象用equals方法比较返回true，Set就不 会接受这两个对象。

**HashSet**  
HashSet有以下特点  
 不能保证元素的排列顺序，顺序有可能发生变化  
 不是同步的  
 集合元素可以是null,但只能放入一个null  
当向HashSet结合中存入一个元素时，HashSet会调用该对象的hashCode()方法来得到该对象的hashCode值，然后根据 hashCode值来决定该对象在HashSet中存储位置。  
简单的说，HashSet集合判断两个元素相等的标准是两个对象通过equals方法比较相等，并且两个对象的hashCode()方法返回值相 等  
注意，如果要把一个对象放入HashSet中，重写该对象对应类的equals方法，也应该重写其hashCode()方法。其规则是如果两个对 象通过equals方法比较返回true时，其hashCode也应该相同。另外，对象中用作equals比较标准的属性，都应该用来计算 hashCode的值。

**LinkedHashSet**

**java.util.LinkedHashSet<E>继承于**[java.util.HashSet](mk:@MSITStore:C:\\Users\\Administrator\\Desktop\\[Java参考文档].JDK_API_1_6_zh_CN.CHM::/java/util/../../java/util/HashSet.html" \o "java.util 中的类)<E>

LinkedHashSet集合同样是根据元素的hashCode值来决定元素的存储位置，但是它同时使用链表维护元素的次序。这样使得元素看起 来像是以插入顺序保存的，也就是说，当遍历该集合时候，LinkedHashSet将会以元素的添加顺序访问集合的元素。  
LinkedHashSet在迭代访问Set中的全部元素时，性能比HashSet好，但是插入时性能稍微逊色于HashSet。

**TreeSet类**  
TreeSet是SortedSet接口的唯一实现类，TreeSet可以确保集合元素处于排序状态。TreeSet支持两种排序方式，自然排序 和定制排序，其中自然排序为默认的排序方式。向TreeSet中加入的应该是同一个类的对象。  
TreeSet判断两个对象不相等的方式是两个对象通过equals方法返回false，或者通过CompareTo方法比较没有返回0  
自然排序  
自然排序使用要排序元素的CompareTo（Object obj）方法来比较元素之间大小关系，然后将元素按照升序排列。

Java提供了一个Comparable接口，该接口里定义了一个compareTo(Object obj)方法，该方法返回一个整数值，实现了该接口的对象就可以比较大小。  
obj1.compareTo(obj2)方法如果返回0，则说明被比较的两个对象相等，如果返回一个正数，则表明obj1大于obj2，如果是 负数，则表明obj1小于obj2。  
如果我们将两个对象的equals方法总是返回true，则这两个对象的compareTo方法返回应该返回0  
定制排序  
自然排序是根据集合元素的大小，以升序排列，如果要定制排序，应该使用Comparator接口，实现 int compare(T o1,T o2)方法