### **一、Comparable简介**

　　Comparable是排序接口。若一个类实现了Comparable接口，就意味着该类支持排序。实现了Comparable接口的类的对象的列表或数组可以通过Collections.sort或Arrays.sort进行自动排序

### **二、Comparator简介**

　　Comparator是比较接口，我们如果需要控制某个类的次序，而该类本身不支持排序(即没有实现Comparable接口)，那么我们就可以建立一个“该类的比较器”来进行排序，这个“比较器”只需要实现Comparator接口即可。也就是说，我们可以通过实现Comparator来新建一个比较器，然后通过这个比较器对类进行排序

**三、区别**

Comparable是排序接口，若一个类实现了Comparable接口，就意味着“该类支持排序”。而Comparator是比较器，我们若需要控制某个类的次序，可以建立一个“该类的比较器”来进行排序。

　　Comparable相当于“内部比较器”，而Comparator相当于“外部比较器”。

两种方法各有优劣， 用Comparable 简单， 只要实现Comparable 接口的对象直接就成为一个可以比较的对象，但是需要修改源代码。 用Comparator 的好处是不需要修改源代码， 而是另外实现一个比较器， 当某个自定义的对象需要作比较的时候，把比较器和对象一起传递过去就可以比大小了， 并且在Comparator 里面用户可以自己实现复杂的可以通用的逻辑，使其可以匹配一些比较简单的对象，那样就可以节省很多重复劳动了。

**红黑树的性质**

1. 每个节点或是红色的，或是黑色的

2. 根节点是黑色的

3.每个叶节点（NIL）是黑色的

4.如果一个节点是红色的，则它的两个子节点都是黑色的

5.对于每个几点，从该节点到其所有后代叶节点的简单路径上，均包含相同数目的黑色节点。