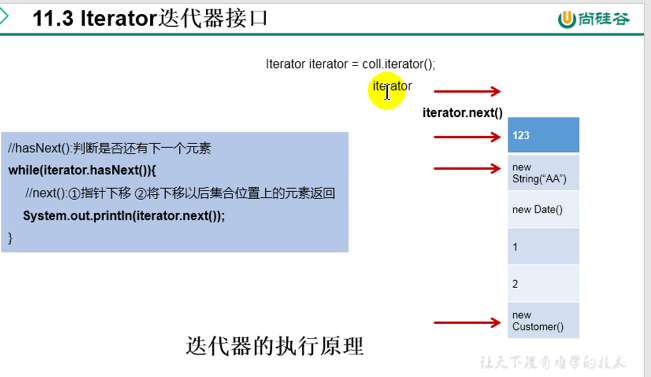
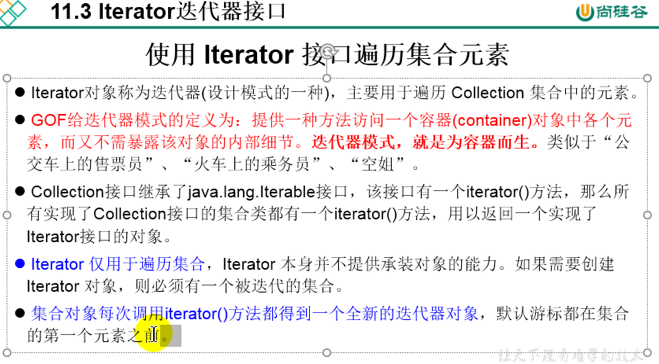
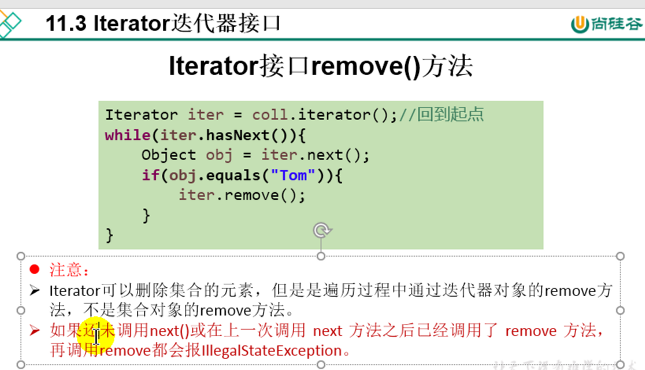


Set接口： 无序性、确定性、互异性

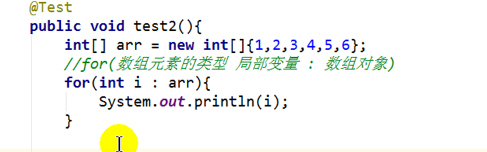
.虽然是无序性，不代表是随机性，她的排序方式是以哈希值进行排序，所以与插入顺序无关。

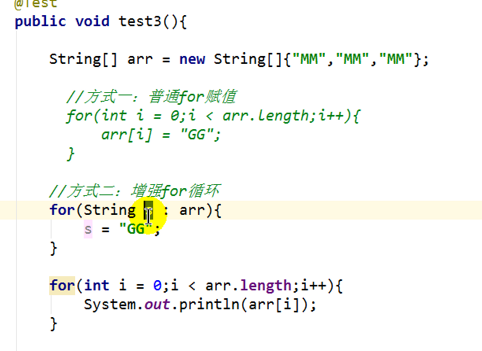




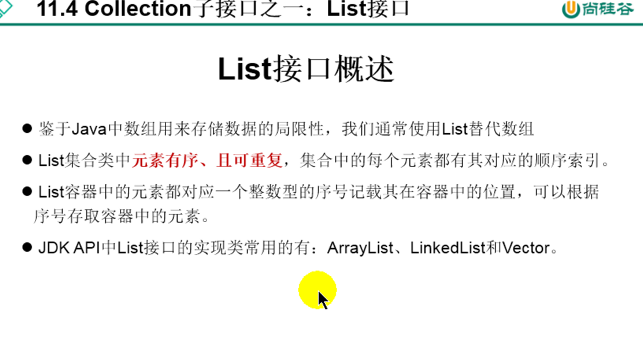








遍历集合和数据的时候使用foreach 中里边进行修改是不能够修改到最外边的集合的数据的，只是局部的一个修改，因为它的效果类似于将arr赋值到一个新的变量里，因此不是修改集合。



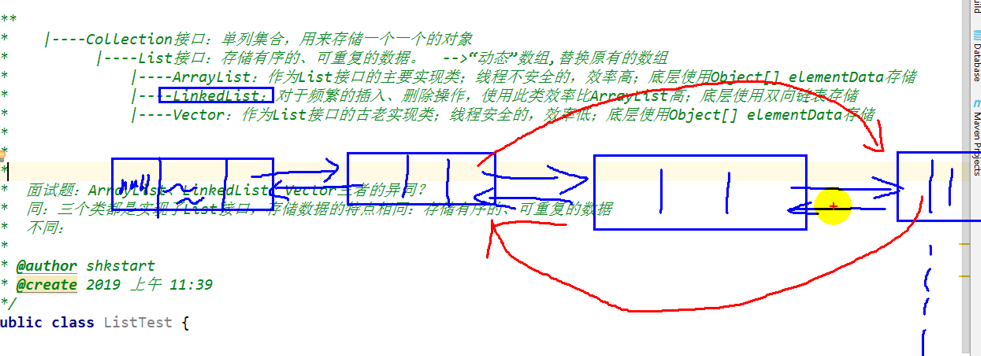


存储都是有序的，可重复的，都是collection接口

Arraylist：线程不安全，效率高

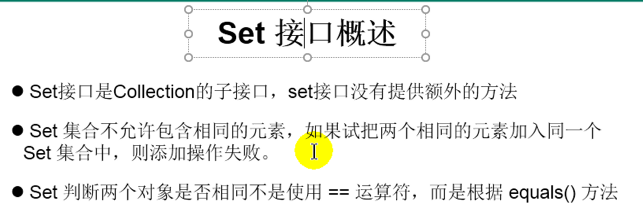
Linkedlist：频繁插入的时候建议使用，效率比ArrayList高，她存储方式是以链表方式进行存储。

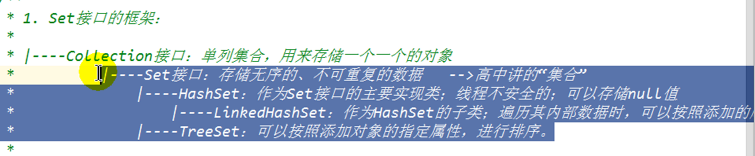
Vector：线程安全，效率低。

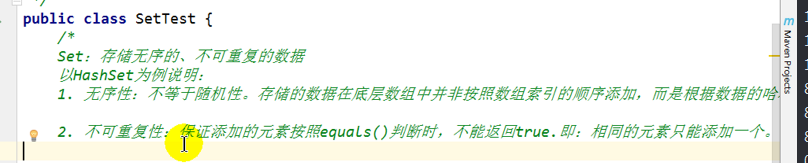


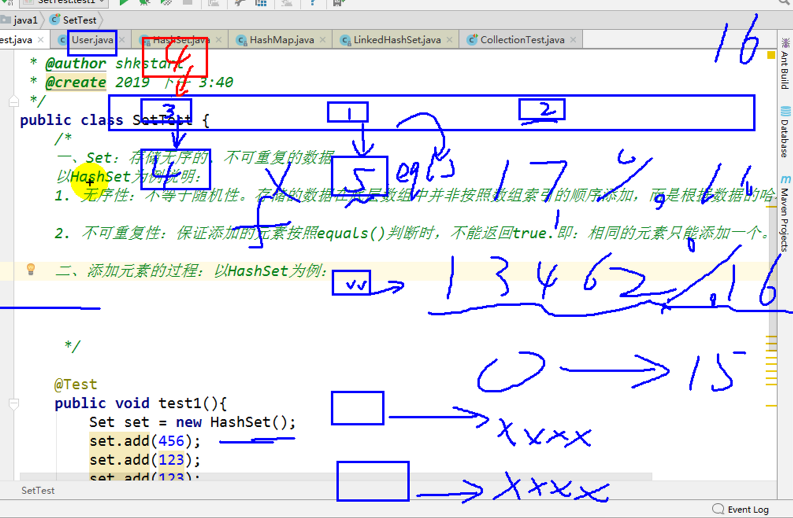
LinkedList:插入与删除 原理： 例如：有三个人，分别为ABC ，A如果要给C发消息，但是他通过B进行一个传话给C；此时 删除操作： C不需要B进行传话了，C直接去找A进行互动，则B就没有他的事情了。 插入操作也一样，A需要给C钱，但是A不想走那么远了，因此将钱给到B ，B再拿给C，因此形成了插入的操作。

对于数据多，删除插入操作频繁的话建议使用LinkedList，因为ArrayList删除或插入操作，他需要将插入的这个位置的后面的元素都一个一个的进行位置的挪动。因此动作大，操作就慢。

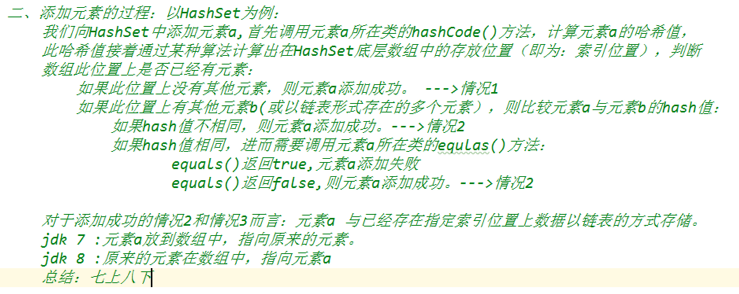


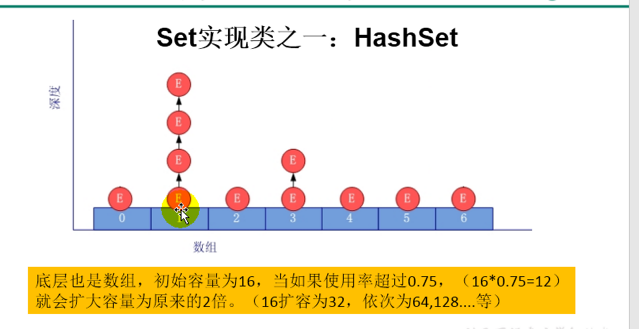


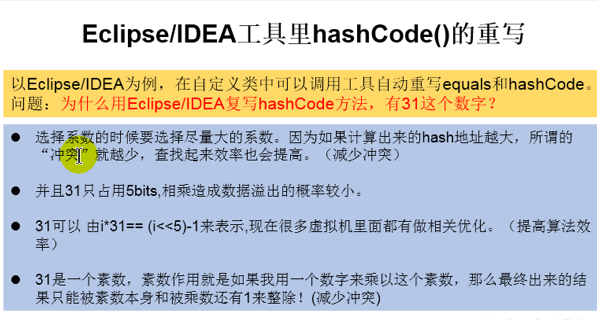


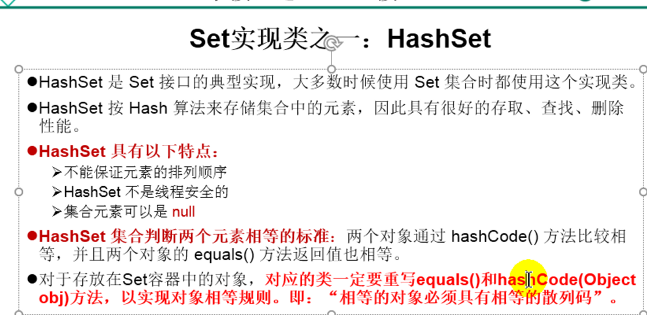


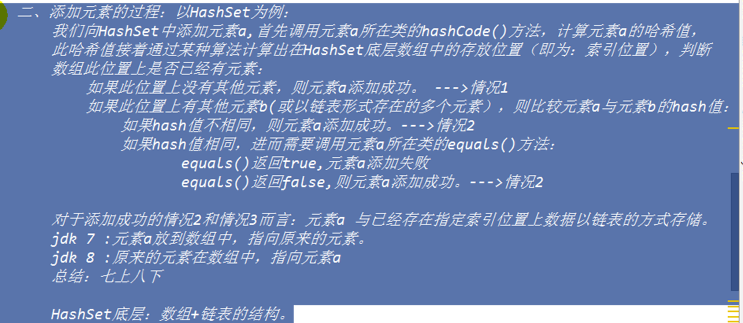
Set接口存放类型如果哈希值一样，他会以列表形式存放，前提是值不相等，如果相等添加失败。



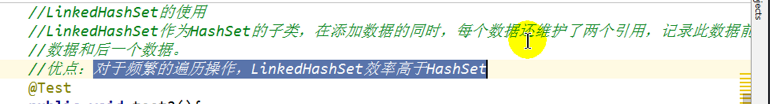






Hashset在面试过程中不会提问！但是会问是由什么来实现的？ 是以hashmap来实现的。所以接下来的hashmap很重要！！！

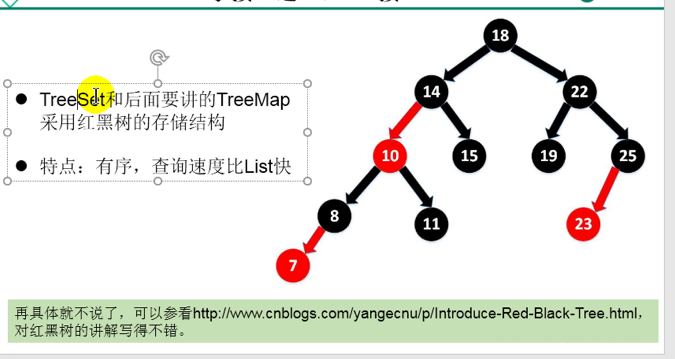


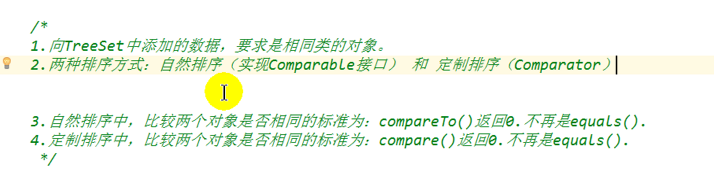


这个方法在实体类中写



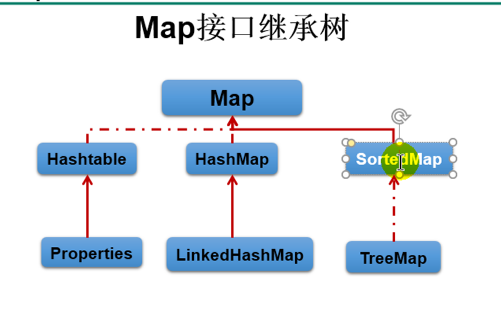
这个是对比treeset插入相同姓名但是年龄不相同的情况下，这要表名是两个不同的对象，因此需要进行一个对比。

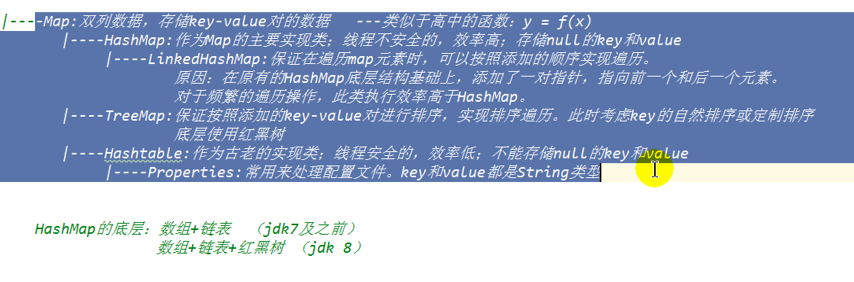


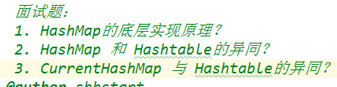
.

定制排序：







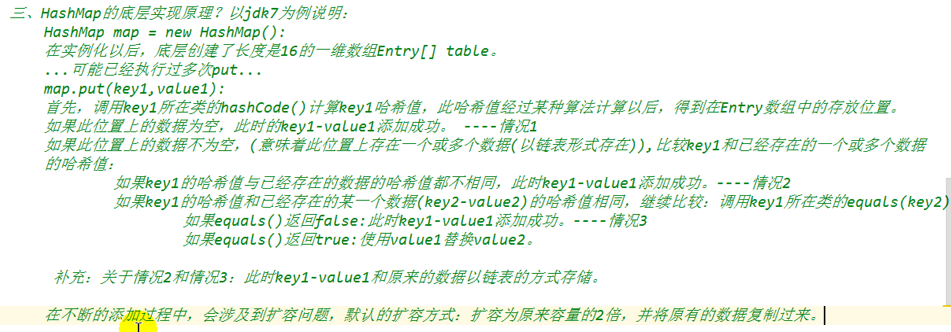


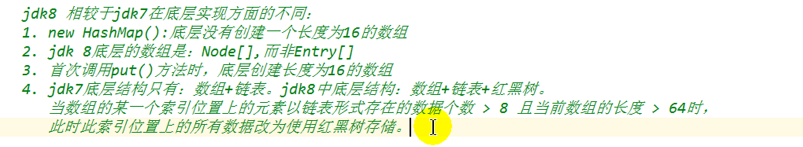
currentHashMap:使用分段锁解决共享数据的高并发，线程不安全的问题。

<https://zhuanlan.zhihu.com/p/31614308> 有详细的介绍！！

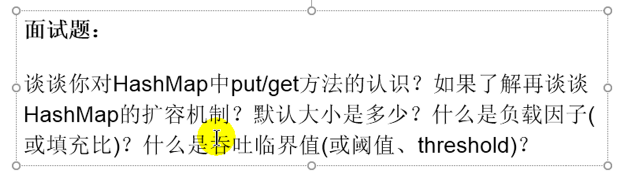
HashMap结构：key用set存储，不可重复，无序性；value用collection存储，可重复，无序性。Key所在的类需要重写equals与hashcode方法。（hashmap为例）；value所在的类需要重写equals方法。

Map中数据都是存放在entry[]，key与value都是被entry包含着的；无序性，不可重复。下面的图片为面试常问问题：



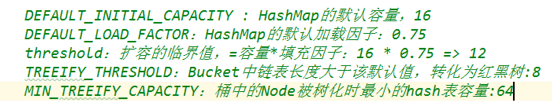


原因是因为数据变多了，使用红黑树遍历会快许多，红黑树原理（比较第一个数据是否比他大，是的话遍历比他大的一边，如果比他小就遍历比他小的一遍，因此会减少遍历的次数）



HashMap：

扩容机制：Hashmap是通过Node[]数组存放键值对的，默认值是16，根据临界值12，到达了临界值之后，判断13或往后的位置是否为空，如果是空，则先不扩容，如果不为空，则扩容到比原来的两倍，在JAVA8中，扩容大于64之后，且存放在数组的某一个索引位置中的元素的个数超出8个，则改为用红黑树方式存储。负载因子默认值是0.75，通过负载因子计算出临界值，决定了数组扩容的时机，如果负载因子大，则存放在数组上某一个索引位置上的以链表形式存放的数据就越多；如果负载因子过小，则Node[]数组的利用率就较低。因此使用默认值0.75相对来说比较合适。













Collection：是存储单列数据的接口

Collections：是一个工具类，是操作collection的工具类

HashSet与TreeSet区别？

Collections工具类中常用的方法：

