List 和 Set 比较，各自的子类比较

对比一：Arraylist 与 LinkedList 的比较

1、ArrayList 是实现了基于动态数组的数据结构,因为地址连续，一旦数据存储好了，查询操作效率会比较高（在内存里是连着放的）。

2、因为地址连续， ArrayList 要移动数据,所以插入和删除操作效率比较低。

3、LinkedList 基于链表的数据结构,地址是任意的，所以在开辟内存空间的时候不需要等一个连续的地址，对于新增和删除操作 add 和 remove，LinedList 比较占优势。

4、因为 LinkedList 要移动指针,所以查询操作性能比较低。

适用场景分析：

当需要对数据进行对此访问的情况下选用 ArrayList，当需要对数据进行多次增加删除修改时采用 LinkedList。

对比二：ArrayList 与 Vector 的比较

1、Vector 的方法都是同步的，是线程安全的，而 ArrayList 的方法不是，由于线程的同步必然要影响性能。因此，ArrayList 的性能比 Vector 好。

2、当 Vector 或 ArrayList 中的元素超过它的初始大小时，Vector 会将它的容量翻倍，而 ArrayList 只增加 50%的大小，这样。ArrayList 就有利于节约内存空间。

3、大多数情况不使用 Vector，因为性能不好，但是它支持线程的同步，即某一时刻只有一个线程能够写 Vector，避免多线程同时写而引起的不一致性。

4、Vector 可以设置增长因子，而 ArrayList 不可以。

适用场景分析：

1、Vector 是线程同步的，所以它也是线程安全的，而 ArrayList 是线程异步的，是不安全的。如果不考虑到线程的安全因素，一般用 ArrayList 效率比较高。

2、如果集合中的元素的数目大于目前集合数组的长度时，在集合中使用数据量比较大的数据，用 Vector 有一定的优势。

对比三：HashSet 与 TreeSet 的比较

1.TreeSet 是二叉树实现的，Treeset 中的数据是自动排好序的，不允许放入 null 值 。

2.HashSet 是哈希表实现的，HashSet 中的数据是无序的，可以放入 null，但只能放入一个 null，两者中的值都不能重复，就如数据库中唯一约束 。

3.HashSet 要求放入的对象必须实现 HashCode()方法，放入的对象，是以 hashcode 码作为标识的，而具有相同内容的 String 对象，hashcode 是一样，所以放入的内容不能重复。但是同一个类的对象可以放入不同的实例。

适用场景分析：

HashSet 是基于 Hash 算法实现的，其性能通常都优于 TreeSet。我们通常都应该使用 HashSet，在我们需要排序的功能时，我们才使用 TreeSet。

HashMap 和 ConcurrentHashMap 的区别

1、HashMap 不是线程安全的，而 ConcurrentHashMap 是线程安全的。

2、ConcurrentHashMap 采用锁分段技术，将整个 Hash 桶进行了分段 segment，也就是将这个大的数组分成了几个小的片段 segment，而且每个小的片段 segment 上面都有锁存在，那么在插入元素的时候就需要先找到应该插入到哪一个片段 segment，然后再在这个片段上面进行插入，而且这里还需要获取 segment 锁。

3、ConcurrentHashMap 让锁的粒度更精细一些，并发性能更好。

至于两者的底层实现，你如果想通过一篇文章就理解了，那就 too young 了，好好找些博文+看源码去吧。

HashTable 和 ConcurrentHashMap 的区别

它们都可以用于多线程的环境，但是当 Hashtable 的大小增加到一定的时候，性能会急剧下降，因为迭代时需要被锁定很长的时间。因为 ConcurrentHashMap 引入了分割(segmentation)，不论它变得多么大，仅仅需要锁定 map 的某个部分，而其它的线程不需要等到迭代完成才能访问 map。简而言之，在迭代的过程中，ConcurrentHashMap 仅仅锁定 map 的某个部分，而 Hashtable 则会锁定整个 map。

String,StringBuffer 和 StringBuilder 的区别

1、运行速度，或者说是执行速度，在这方面运行速度快慢为：StringBuilder > StringBuffer > String。

2、线程安全上，StringBuilder 是线程不安全的，而 StringBuffer 是线程安全的。

适用场景分析：

String：适用于少量的字符串操作的情况

StringBuilder：适用于单线程下在字符缓冲区进行大量操作的情况

StringBuffer：适用多线程下在字符缓冲区进行大量操作的情况

wait 和 sleep 的区别

1、sleep()方法是属于 Thread 类中的，而 wait()方法，则是属于 Object 类中的。

2、sleep()方法导致了程序暂停执行指定的时间，让出 cpu 给其他线程，但是他的监控状态依然保持着，当指定的时间到了又会自动恢复运行状态。所以在调用 sleep()方法的过程中，线程不会释放对象锁。

3、调用 wait()方法的时候，线程会放弃对象锁，进入等待此对象的等待锁定池，只有针对此对象调用 notify()方法后本线程才进入对象锁定池准备获取对象锁进入运行状态。