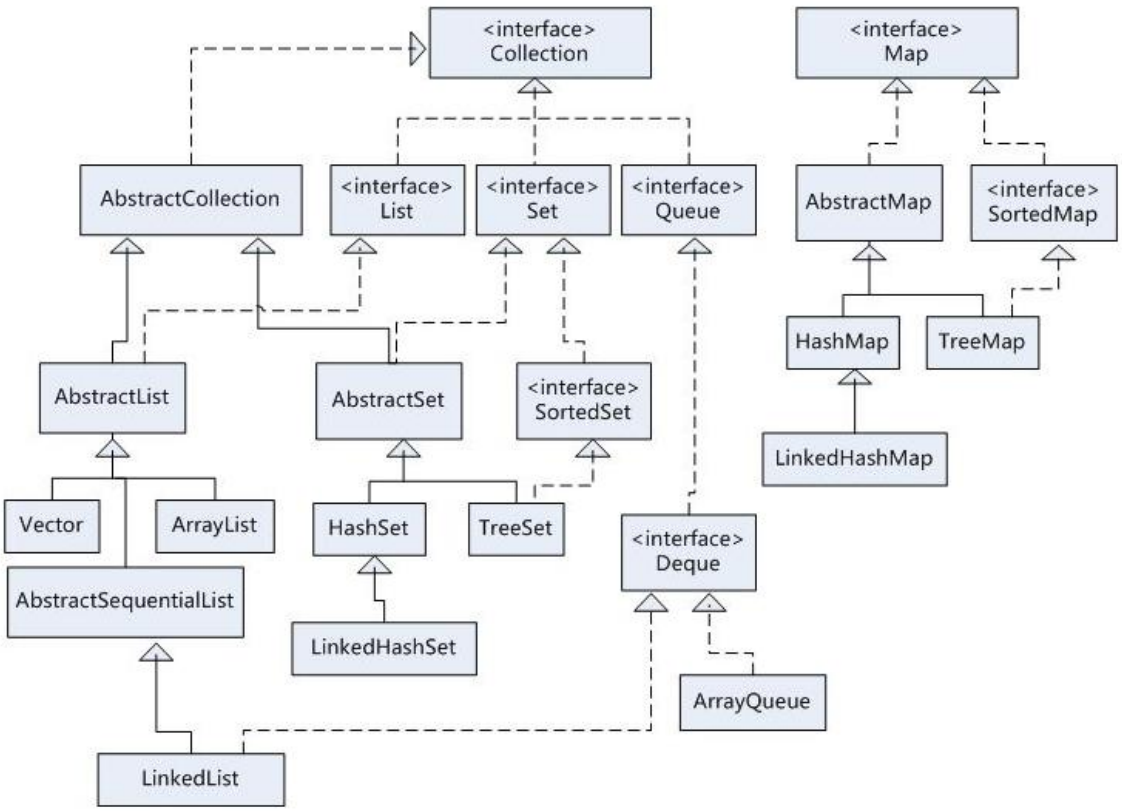

1、Collection: 单列集合（需要保存若干个对象的时候使用集合）

Collection 层次结构中的根接口。Collection 表示一组对象，这些对象也称为 collection 的元素。一些 collection 允许有重复的元素，而另一些则不允许。一些 collection 是有序的，而另一些则是无序的。JDK 不提供此接口的任何直接 实现：它提供更具体的子接口（如 Set 和 List）实现。此接口通常用来传递 collection，并在需要最大普遍性的地方操作这些 collection。

1.1、面试题

Collection 和 Collections 的区别？

Collection 是集合类的上级接口,继承与他的接口主要有 Set 和 List.

Collections 是针对集合类的一个帮助类,他提供一系列静态方法实现对各种集合的搜索、排序、线程安全化等操作

2、list（有存储顺序, 可重复）

ArrayList：数组实现，查找快，增删慢。

数组为什么是查询快?因为数组的内存空间地址是连续的。

ArrayList 底层维护了一个 Object[] 用于存储对象，（源码可查）默认数组的长度是 10。可以通过 new ArrayList(15)显式的指定用于存储对象的数组的长度。

当默认的或者指定的容量不够存储对象的时候，容量自动增长为原来的容量的 1.5 倍。

由于 ArrayList 是数组实现，在增和删的时候会牵扯到数组增容， 以及拷贝元素， 所以慢。数组是可以直接按索引查找，所以查找时较快。

可以考虑，假设向数组的 0 角标未知添加元素，那么原来的角标位置的元素需要整体往后移，并且数组可能还要增容，一旦增容，就需要要将老数组的内容拷贝到新数组中，所以数组的增删的效率是很低的。

2.1、面试题

Array 和 ArrayList 有何区别？什么时候更适合用 Array？

1. Array 可以容纳基本类型和对象，而 ArrayList 只能容纳对象。
2. Array 是指定大小的，而 ArrayList 大小是固定的

LinkedList：链表实现，增删快，查找慢

由于 LinkedList 在内存中的地址不连续，是链表实现的，增加时只要让前一个元素记住自己就可以， 删除时让前一个元素记住后一个元素，后一个元素记住前一个元素。 这样的增删效率较高。但查询时需要一个一个的遍历，所以效率较低。

2.1、ArrayList 和 LinkedList 的存储查找的优缺点？

1、ArrayList 是采用动态数组来存储元素的，它允许直接用下标号来直接查找对应的元素。但是，但是插入元素要涉及数组元素移动及内存的操作。

总结：查找快，增删慢。

2、LinkedList 是采用双向链表实现存储，按序号索引数据需要进行前向或后向遍历，但是插入数据时只需要记录本项的前后项即可，所以插入速度较快。

总结：增删快，查找慢。

Vector：和ArrayList原理相同，但线程安全，效率略低。和ArrayList实现方

式相同， 但考虑了线程安全问题，所以效率略低。

3、set（无存储顺序, 不可重复）

HashSet：哈希表边存放的是哈希值。HashSet 存储元素的顺序并不是按照存入时的顺序（和 List 显然不同） 是按照哈希值来存的所以取数据也是按照哈希值取得。HashSet 不存入重复元素的规则.使用 hashcode 和 equals。

由于 Set 集合是不能存入重复元素的集合。那么 HashSet 也是具备这一特性的。HashSet 如何检查重复？HashSet 会通过元素的 hashcode（ ）和 equals 方法进行判断元素师否重复。

博客地址：http://blog.csdn.net/javawebrookie/article/details/49028263(Andy)

面试题

hashCode()和 equals()方法有何重要性？

HashMap 使用 Key 对象的 hashCode()和 equals()方法去决定 key-value 对的索引。当我们试着从 HashMap 中获取值的时候，这些方法也会被用到。如果这些方法没有被正确地实现，在这种情况下，两个不同 Key 也许会产生相同的 hashCode()和 equals()输出，HashMap 将会认为它们是相同的，然后覆盖它们，而非把它们存储到不同的地方。同样的，所有不允许存储重复数据的集合类都使用 hashCode()和 equals()去查找重复，所以正确实现它们非常重要。equals()和 hashCode()的实现应该遵循以下规则：

（1）如果 o1.equals(o2)，那么 o1.hashCode() == o2.hashCode()总是为 true 的。

（2）如果 o1.hashCode() == o2.hashCode()，并不意味着 o1.equals(o2)会为 true。

TreeSet ：红-黑树的数据结构（特定类型的二叉树），默认对元素进行自然排序（String）。如果在比较的时候两个对象返回值为 0，那么元素重复。

问题：为什么使用 TreeSet 存入字符串，字符串默认输出是按升序排列的?

因为字符串实现了一个接口，叫做 Comparable 接口，字符串重写了该接口的 compareTo 方法，所以 String 对象具备了比较性，那么同样道理,我的自定义元素(例如 Person 类,Book 类)想要存入 TreeSet 集合,就需要实现该接

口，也就是要让自定义对象具备比较性。存入 TreeSet 集合中的元素要具备比较性。

比较性要实现 Comparable 接口，重写该接口的 compareTo 方法 TreeSet 属于 Set 集合，该集合的元素是不能重复的，TreeSet 如何保证元素的唯一性。通过 compareTo 或者 compare 方法中的来保证元素的唯一性。添加的元素必须要实现 Comparable 接口。当 compareTo()函数返回值为 0 时，说明两个对象相等，此时该对象不会添加进来。

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。

4、Map（键值对）

Map 是将键映射到值的对象。一个映射不能包含重复的键；每个键最多只能映射到一个值。

面试题

为何 Map 接口不继承 Collection 接口？

尽管 Map 接口和它的实现也是集合框架的一部分，但 Map 不是集合，集合也不是 Map。因此，Map 继承 Collection 毫无意义，反之亦然。

如果 Map 继承 Collection 接口，那么元素去哪儿？Map 包含 key-value 对，它提供抽取 key 或 value 列表集合的方法，但是它不适合“一组对象”规范。

HashMap：底层是哈希表数据结构，线程是不同步的，可以存入 null 键，null 值。要保证键的唯一性，需要覆盖 hashCode 方法，和 equals 方法。

4.1、HashMap 的面试题？

什么是 HashMap？你为什么用到它？

譬如 HashMap 可以接受 null 键值和值，而 Hashtable 则不能；HashMap 是非 synchronized；HashMap 很快；以及 HashMap 储存的是键

值对等等。

你知道 HashMap 的工作原理吗？

HashMap 是基于 hashing（散列法）的原理，我们使用 put(key, value)存储对象到 HashMap 中，使用 get(key)从 HashMap 中获取对象。当我们给 put()方法传递键和值时，我们先对键调用 hashCode()方法，返回的 hashCode 用于找到 bucket 位置来储存 Entry 对象。这里关键点在于指出，HashMap 是在 bucket 中储存键对象和值对象，作为 Map.Entry。

当两个对象的 hashcode 相同会发生什么？

因为 hashcode 相同，所以它们的 bucket 位置相同，‘碰撞’会发生。因为 HashMap 使用链表存储对象，这个 Entry(包含有键值对的 Map.Entry 对象)会存储在链表中。”这个答案非常的合理，虽然有很多种处理碰撞的方法，这种方法是最简单的，也正是 HashMap 的处理方法。

如果两个键的 hashcode 相同，你如何获取值对象？

当我们调用 get()方法，HashMap 会使用键对象的 hashcode 找到 bucket 位置，然后获取值对象。如果有两个值对象储存在同一个 bucket，那么将会遍历链表直到找到值对象。

如果 HashMap 的大小超过了负载因子(load factor)定义的容量，怎么办？

默认的负载因子大小为 0.75，也就是说，当一个 map 填满了 75%的 bucket 时候，和其它集合类(如 ArrayList 等)一样，将会创建原来 HashMap 大小的两倍的 bucket 数组，来重新调整 map 的大小，并将原来的对象放入新的 bucket 数组中。这个过程叫作 rehashing，因为它调用 hash 方法找到新的 bucket 位置。

你了解重新调整 HashMap 大小存在什么问题吗？

当重新调整 HashMap 大小的时候，确实存在条件竞争，因为如果两个线程都发现 HashMap 需要重新调整大小了，它们会同时试着调整大小。在调整大小的过程中，存储在链表中的元素的次序会反过来，因为移动到新的 bucket 位置的时候，HashMap 并不会将元素放在链表的尾部，而是放在头部，这是为了避免尾部遍历(tail traversing)。如果条件竞争发生了，那么就死循环了。这个时候，你可以质问面试官，为什么这么奇怪，要在多线程的环境下使用

HashMap 呢？

TreeMap：底层是二叉树数据结构。可以对 map 集合中的键进行排序。需要使用 Comparable 或者 Comparator 进行比较排序。return 0，来判断键的唯一性。

5、Iterable

Jdk1.5 之后添加的新接口, Collection 的父接口. 实现了 Iterable 的类就是可迭代的.并且支持增强 for 循环。

面试题

Enumeration 和 Iterator 接口的区别？

Enumeration 的速度是 Iterator 的两倍，也使用更少的内存。Enumeration 是非常基础的，也满足了基础的需要。但是，与 Enumeration 相比，Iterator 更加安全，因为当一个集合正在被遍历的时候，它会阻止其它线程去修改集合。

迭代器取代了 Java 集合框架中的 Enumeration。迭代器允许调用者从集合中移除元素，而 Enumeration 不能做到。为了使它的功能更加清晰，迭代器方法名已经经过改善。

博客地址：http://blog.csdn.net/javawebrookie/article/details/49028263(Andy)