**集合类的一些主要方法和比较**

1. 数组与集合最大的区别就是数组长度空间是固定的，而集合是动态开辟空间来进行存储，但集合只能存储对象的引用。集合主要分为Collection和Map两大类。
2. Collection（单列集合）类主要分为三大类
3. List(链表||线性)：有存储顺序，且存储的值可重复

a2. LinkedList: 链表 a1. ArrayList: 数组实现，所以查找快，删除慢

实现，增删快，查找慢

a3. vector：和ArrayList原理实现相同，即数组实现考虑了线程安全，所以效率低。

1. Set: 无序存储，不可重复

a1. Hashset：不需要排序，效率相对TreeSet效率较高

a2. Treeset： 对元素进行排序

a3. LinkedHashSet：保留了存储顺序，能过滤重复元素

1. Queue(队列)

补充：collection接口的共性方法

增加：

add() 将指定对象存储到容器中，add方法的参数类型是Object ，接

收任意对象

addAll() 将指定集合中的元素添加到调用该方法和集合中

删除：

remove() 将指定的对象从集合中删除

removeAll() 将指定集合中的元素删除

修改：

clear() 清空集合中的所有元素

判断：

isEmpty() 判断集合是否为空

contains() 判断集合何中是否包含指定对象

containsAll() 判断集合中是否包含指定集合，使用equals()判断两个对象是否相等

获取:

int size()返回集合容器的大小

转成数组：

toArray()集合转换数组

1. Map(键值对)主要实现类分为三大类
2. HashMap

特点：保存元素时先进后出，且存储是无序的，查询效率比较高，存储的键值

对可为空，但只能有一个。不支持线程同步，也就是说多线程同时写

HashMap，可能导致数据不一致，也就导致线程不安全。也经常和

Hashtable（键值对不可为空，线程安全）进行比较

1. LinkedHashMap

特点：内部是双向链表结构，保存了元素插入顺序，Iterator遍历是按照元素插入顺序排列的，支持线程同步，线程安全

1. TreeMap

特点：保存键值对是不能为null，自动排序，默认升序，允许键值对重复，遍历元素时随机排列

补充：void clear()从此映射中移除所有映射关系（可选操作）。

boolean containsKey(Object key)如果此映射包含指定键的映射关系，则返回true。

　　 boolean containsValue(Object value)如果此映射将一个或多个键映射到指定值，则返回 true。

　　 Set<Map.Entry<K,V>> entrySet()返回此映射中包含的映射关系的 Set 视图。

　　 boolean equals(Object o)比较指定的对象与此映射是否相等。

　　 V get(Object key)返回指定键所映射的值；如果此映射不包含该键的映射关系，则返回 null。

　　 Int hashCode()返回此映射的哈希码值。

　　 boolean isEmpty()如果此映射未包含键-值映射关系，则返回 true。

　　 Set<K> keySet()返回此映射中包含的键的 Set 视图。

　　 V put(K key, V value)将指定的值与此映射中的指定键关联（可选操作）。

　　 void putAll(Map<? extends K,? extends V> m)从指定映射中将所有映射关系复制到此映射中（可选操作）。

　　 V remove(Object key)如果存在一个键的映射关系，则将其从此映射中移除（可选操作）。

　　 int size()返回此映射中的键-值映射关系数。

　　 Collection<V> values()返回此映射中包含的值的 Collection 视图。

1. 装箱和拆箱：Java装箱操作给每种基本数据类型提供对应的包装器类型,如Integer i=10；

拆箱操作无疑就是将包装器类型的数据转换为基本数据类型