

大家好，很高兴我们可以继续学习交流Java高频面试题。本小节是Java基础篇章的第五小节，在上一小节中，我们将三大集合引出，并且重点介绍了HashMap，Hashtable和ConcurrentHashMap的相关知识点。本小节中，我们继续交流Java中其余的集合知识点，希望大家可以熟练掌握。

17

## （1）TreeMap有哪些特性？

**答：**TreeMap底层使用**红黑树**实现，TreeMap中存储的键值对**按照键来排序**。

- 如果Key存入的是字符串等类型，那么会按照字典默认顺序排序
- 如果传入的是自定义引用类型，比如说User，那么该对象必须实现Comparable接口，并且覆盖其compareTo方法；或者在创建TreeMap的时候，我们必须指定使用的比较器。如下所示：

```
1 // 方式一：定义该类的时候，就指定比较规则
2 class User implements Comparable{
3     @Override
4     public int compareTo(Object o) {
5         // 在这里边定义其比较规则
6         return 0;
7     }
8 }
9 public static void main(String[] args) {
10    // 方式二：创建TreeMap的时候，可以指定比较规则
11    new TreeMap<User, Integer>(new Comparator<User>() {
12        @Override
13        public int compare(User o1, User o2) {
14            // 在这里边定义其比较规则
15            return 0;
16        }
17    });
18 }
```

**解析：**

关于TreeMap的考察，会涉及到两个接口Comparable和Comparator的比较。**Comparable接口的后缀是able大概表示可以的意思**，也就是说一个类如果实现了这个接口，那么这个**类就是可以比较的**。类似的还有cloneable接口表示可以克隆的。而**Comparator则是一个比较器**，是创建TreeMap的时候传入，用来指定比较规则。

## 那么Comparable接口和Comparator接口有哪些区别呢？

- Comparable实现比较简单，但是当需要重新定义比较规则的时候，**必须修改源代码**，即修改User类里边的compareTo方法
- Comparator接口不需要修改源代码，只需要在创建TreeMap的时候**重新传入一个具有指定规则的比较器**即可。

## （2）ArrayList和LinkedList有哪些区别？

**答：**常用的ArrayList和LinkedList的区别总结如下。

- ArrayList底层使用了**动态数组**实现，实质上是一个动态数组
- LinkedList底层使用了**双向链表**实现，可当作堆栈、队列、双端队列使用
- ArrayList在**随机存取**方面效率高于LinkedList

- LinkedList在节点的**增删方面**效率高于ArrayList
- ArrayList必须预留一定的空间，当空间不足的时候，会进行扩容操作
- LinkedList的开销是必须存储节点的信息以及节点的指针信息

17

**解析：**

该题是集合中最常见和最基础的题目之一，List集合也是我们平时使用很多的集合。List接口的常见实现就算ArrayList和LinkedList，我们必须熟练掌握其底层实现以及一些特性。其实还有一个集合**Vector**，它是**线程安全的ArrayList**，但是已经被废弃，不推荐使用了。多线程环境下，我们可以使用CopyOnWriteArrayList替代ArrayList来保证线程安全。

### (3) HashSet和TreeSet有哪些区别？

**答：**HashSet和TreeSet的区别总结如下。

- **HashSet底层使用了Hash表实现。**

保证元素唯一性的原理：判断元素的hashCode值是否相同。如果相同，还会继续判断元素的equals方法，是否为true

- **TreeSet底层使用了红黑树来实现。**

保证元素唯一性是通过Comparable或者Comparator接口实现

**解析：**

其实，HashSet的底层实现还是HashMap，只不过其只使用了其中的Key，具体如下所示：

- HashSet的add方法底层使用HashMap的put方法将key = e, value=PRESENT构建成key-value键值对，当此e存在于HashMap的key中，则value将会覆盖原有value，但是key保持不变，所以如果将一个已经存在的e元素添加到HashSet中，新添加的元素是不会保存到HashMap中，所以这就满足了HashSet中元素不会重复的特性。
- HashSet的contains方法使用HashMap的containsKey方法实现

### (4) LinkedHashMap和LinkedHashSet有了解吗？

**答：**LinkedHashMap在面试题中还是比较常见的。**LinkedHashMap可以记录下元素的插入顺序和访问顺序**，具体实现如下：

- LinkedHashMap内部的Entry继承于HashMap.Node，这两个类都实现了Map.Entry<K,V>
- LinkedHashMap的Entry不光有value，next，还有before和after属性，这样通过一个双向链表，**保证了各个元素的插入顺序**
- 通过构造方法public LinkedHashMap(int initialCapacity,float loadFactor,boolean accessOrder)，**accessOrder传入true可以实现LRU缓存算法（访问顺序）**
- **LinkedHashSet 底层使用LinkedHashMap实现**，两者的关系类似与HashMap和HashSet的关系，大家可以自行类比。

**扩展：什么是LRU算法？LinkedHashMap如何实现LRU算法？**

**LRU (Least recently used, 最近最少使用) 算法**根据数据的历史访问记录来进行淘汰数据，其核心思想是“如果数据最近被访问过，那么将来被访问的几率也更高”。

由于LinkedHashMap可以记录下Map中元素的访问顺序，所以可以轻易的实现LRU算法。只需要**将构造方法的accessOrder传入true，并且重写removeEldestEntry方法**即可。具体实现参考如下：

```
1 package pak2;
2
3 import java.util.LinkedHashMap;
4 import java.util.Map;
5
6 public class LRUTest {
7
8     private static int size = 5;
9
10    public static void main(String[] args) {
11        Map<String, String> map = new LinkedHashMap<String, String>(size, 0.75f, true) {
12            @Override
13            protected boolean removeEldestEntry(Map.Entry<String, String> eldest) {
14                return size() > size;
15            }
16        };
17        map.put("1", "1");
18        map.put("2", "2");
19        map.put("3", "3");
20        map.put("4", "4");
21        map.put("5", "5");
22        System.out.println(map.toString());
23
24        map.put("6", "6");
25        System.out.println(map.toString());
26        map.get("3");
27        System.out.println(map.toString());
28        map.put("7", "7");
29        System.out.println(map.toString());
30        map.get("5");
31        System.out.println(map.toString());
32    }
33 }
```

17

## (5) List和Set的区别？

答：List和Set的区别可以简单总结如下。

- List是有序的并且元素是**可以重复**的
- Set是无序（LinkedHashSet除外）的，并且元素是**不可以重复**的  
(此处的有序和无序是指**放入顺序和取出顺序**是否保持一致)

## (6) Iterator和ListIterator的区别是什么？

答：常见的两种迭代器的区别如下。

- Iterator可以遍历list和set集合；ListIterator只能用来遍历list集合
- Iterator前者只能前向遍历集合；ListIterator可以前向和后向遍历集合
- ListIterator其实就是实现了前者，并且增加了一些新的功能。

解析：

Iterator其实就是一个迭代器，在遍历集合的时候需要使用。Demo实现如下：

```
1 ArrayList<String> list = new ArrayList<>();
2 list.add("zhangsan");
3 list.add("lisi");
4 list.add("yangwenqiang");
5 // 创建迭代器实现遍历集合
6 Iterator<String> iterator = list.iterator();
7 while(iterator.hasNext()){
8     System.out.println(iterator.next());
9 }
```

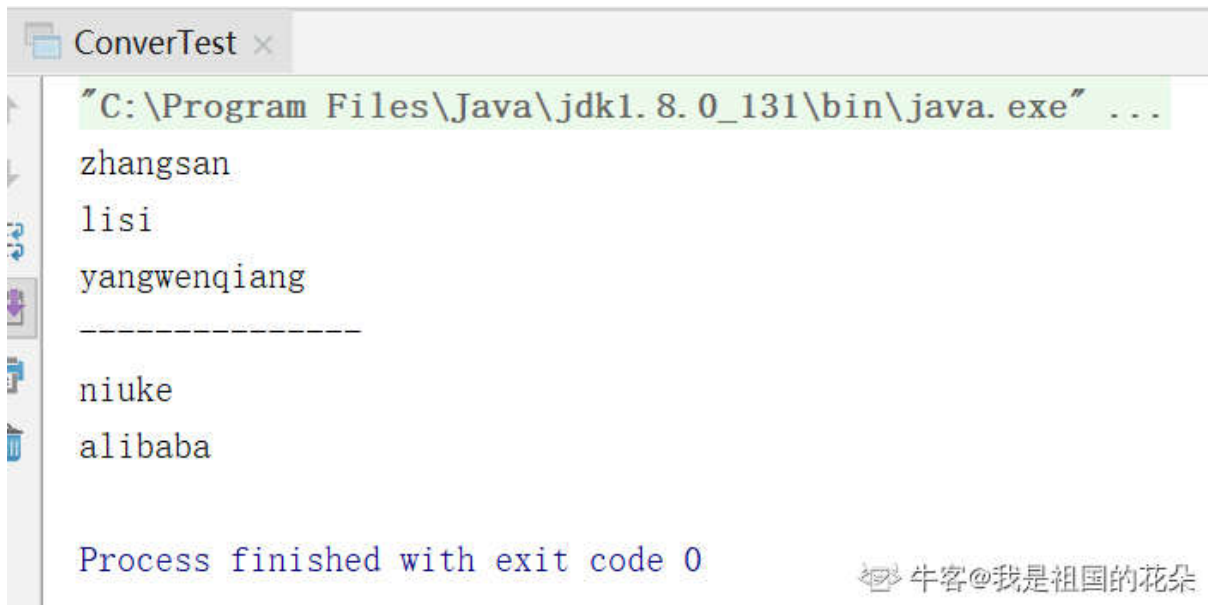
17

## (7) 数组和集合List之间的转换:

答：数组和集合List的转换在我们的日常开发中是很常见的一种操作，主要通过**Arrays.asList**以及**List.toArray**方法来搞定。这里给出Demo演示：

```
1 package niuke;
2
3 import java.util.ArrayList;
4 import java.util.Arrays;
5 import java.util.List;
6
7 public class ConverTest {
8     public static void main(String[] args) {
9         // list集合转换成数组
10        ArrayList<String> list = new ArrayList<>();
11        list.add("zhangsan");
12        list.add("lisi");
13        list.add("yangwenqiang");
14        Object[] arr = list.toArray();
15        for (int i = 0; i < arr.length; i++) {
16            System.out.println(arr[i]);
17        }
18        System.out.println("-----");
19        // 数组转换为list集合
20        String[] arr2 = {"niuke", "alibaba"};
21        List<String> asList = Arrays.asList(arr2);
22        for (int i = 0; i < asList.size(); i++) {
23            System.out.println(asList.get(i));
24        }
25    }
26 }
27 }
```

输出结果如下：



```
ConverTest x
"C:\Program Files\Java\jdk1.8.0_131\bin\java.exe" ...
zhangsan
lisi
yangwenqiang
-----
niuke
alibaba

Process finished with exit code 0
牛客@我是祖国的花朵
```

17

### 解析:

关于数组和集合之间的转换是一个常用操作，这里主要讲解几个需要注意的地方吧。

### 数组转为集合List:

通过Arrays.asList方法搞定，转换之后不可以使用add/remove等修改集合的相关方法，因为该方法返回的**其实是一个Arrays的内部私有的一个类ArrayList**，该类继承于AbstractList，并没有实现这些操作方法，调用将会直接抛出UnsupportedOperationException异常。这种转换体现的是一种**适配器模式**，只是转换接口，本质上还是一个数组。

### 集合转换数组:

List.toArray方法搞定了集合转换成数组，这里**最好传入一个类型一样的数组**，大小就是list.size()。因为如果入参分配的数组空间不够大时，toArray方法内部将重新分配内存空间，并返回新数组地址；如果数组元素个数大于实际所需，下标为list.size()及其之后的数组元素将被置为null，其它数组元素保持原值。所以，建议该方法入参数组的大小与集合元素个数保持一致。

若是**直接使用toArray无参方法**，此方法返回值只能是Object[]类，若强转其它类型数组将出现ClassCastException错误。

## (8) Collection和Collections有什么关系?

**答:** 这是Java中的一类问题，类似的还有Array和Arrays，Executor和Executors有什么区别与联系? 聪明的你可以总结一下吗? 知道答案的同学可以在评论区留言，来帮助更多的同学吧。

### 总结:

本小节中，我们交流学习了Java基础中的三大集合，集合的重要性不言而喻，几乎是面试中的必考知识点。这里给出建议，有精力和能力的同学可以打开JDK的源码，好好熟悉下常见集合类的实现方式。当然如果你遇到问题，可以在评论区留言，我们可以一起探讨学习，一起进步。

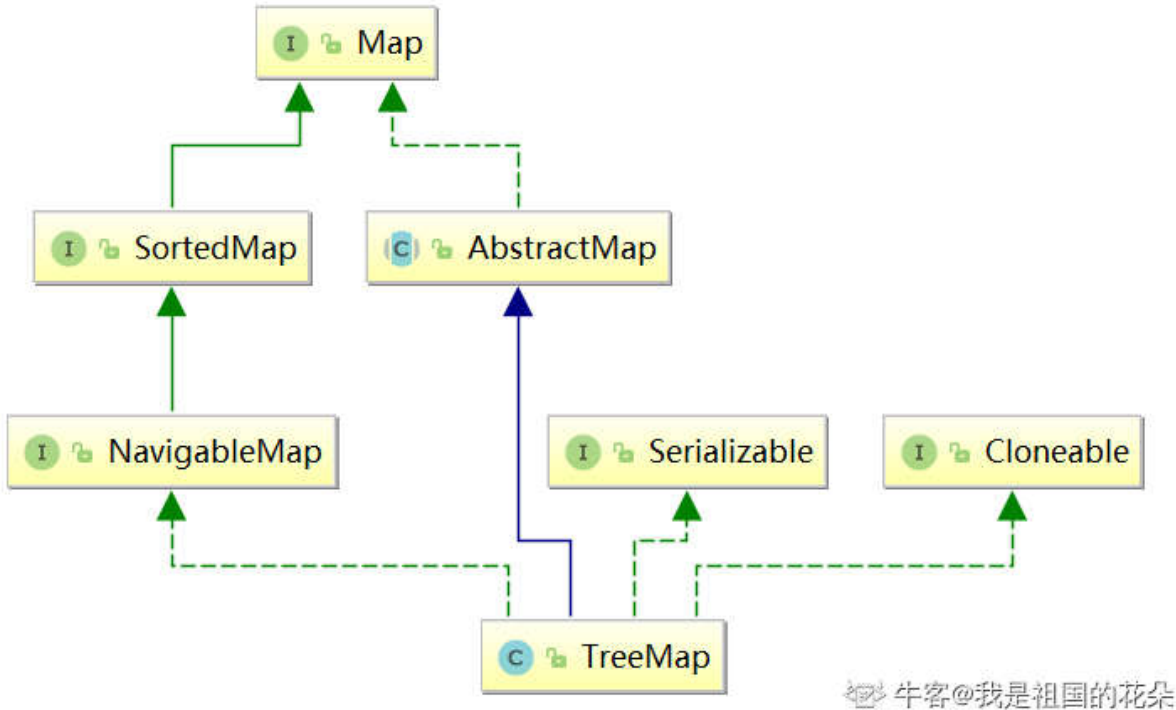
限于作者水平，文章中难免会有不妥之处。大家在学习过程中遇到我没有表达清楚或者表述有误的地方，欢迎随时在文章下边指出，我会及时关注，随时改正。另外，大家有任何话题都可以在下边留言，我们一起交流探讨。

附图：

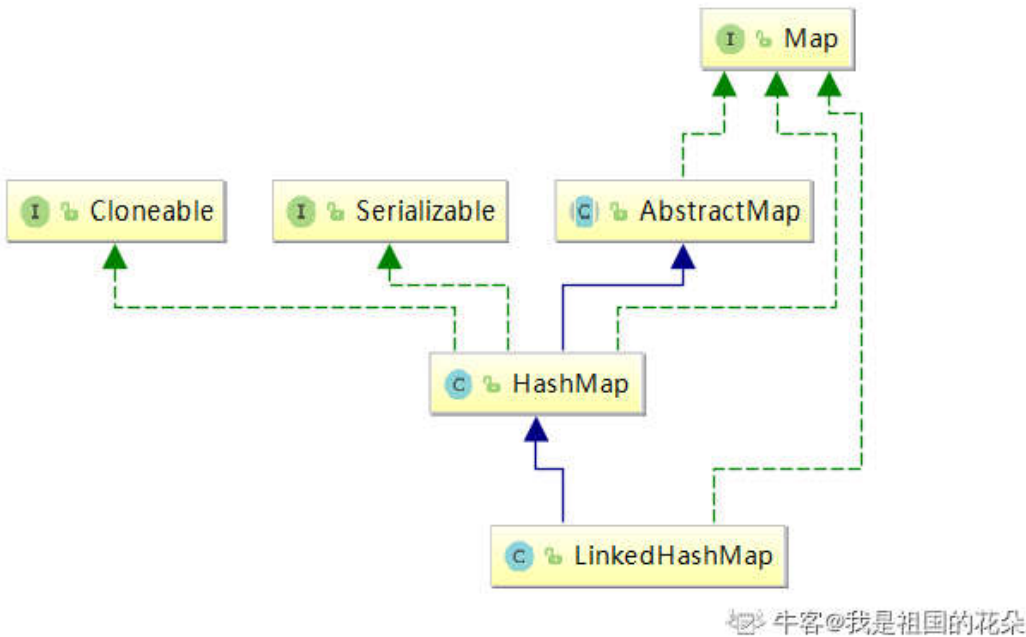
集合的类图：

我们接着给出本节所涉及到的集合的类图结构：（C表示这是一个类，I表示这是一个接口）

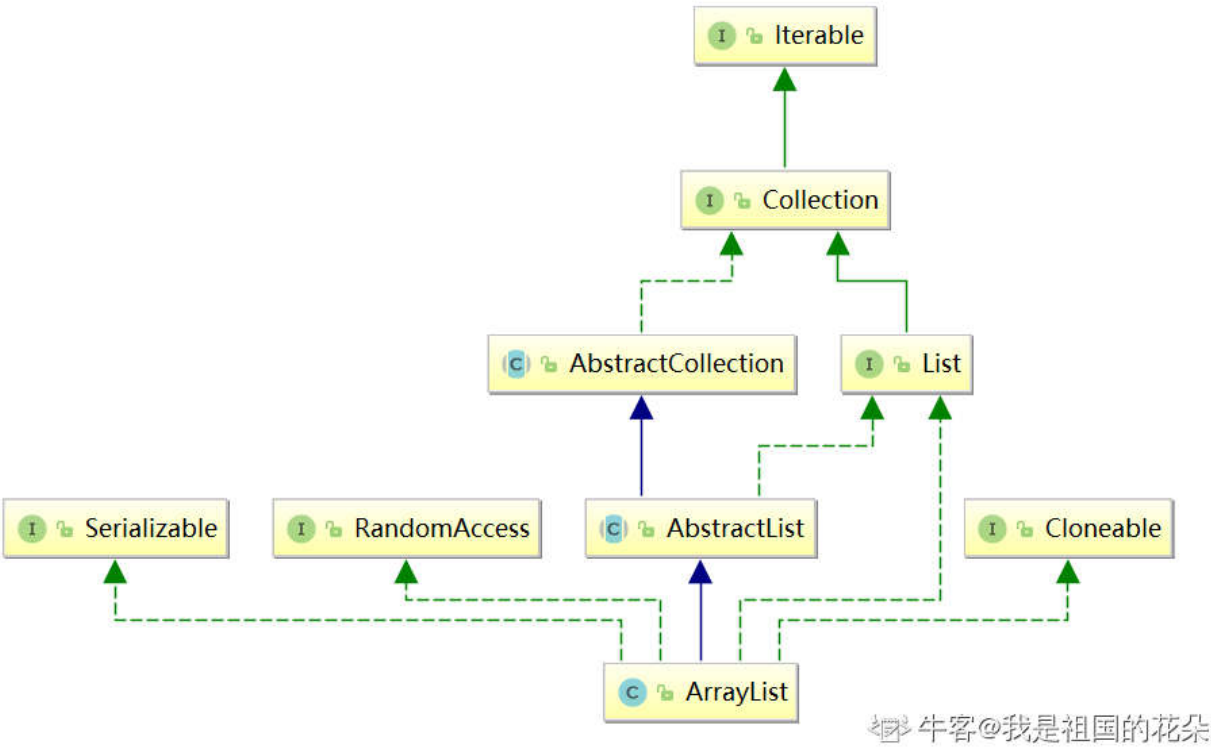
- **TreeMap**的类图结构：



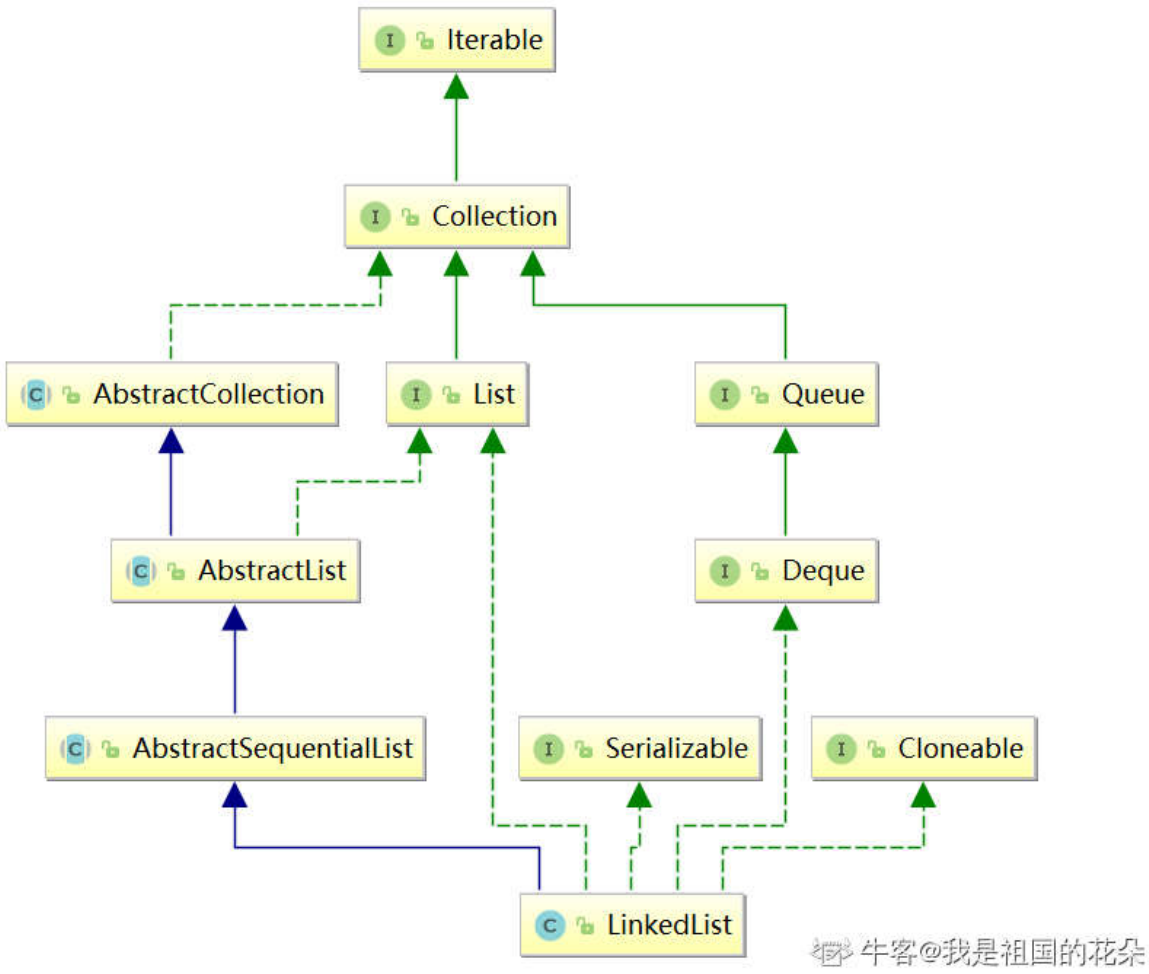
- **LinkedHashMap**的类图结构：



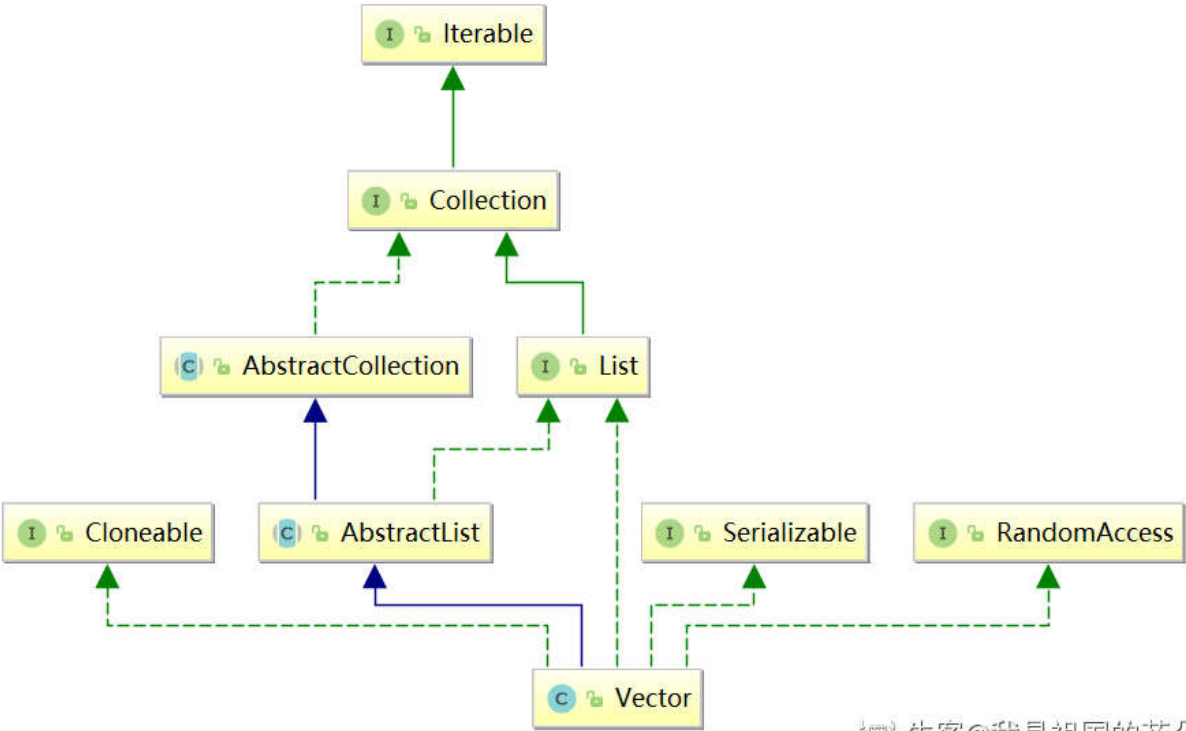
- **ArrayList**的类图结构：



• `LinkedList`的类图结构:

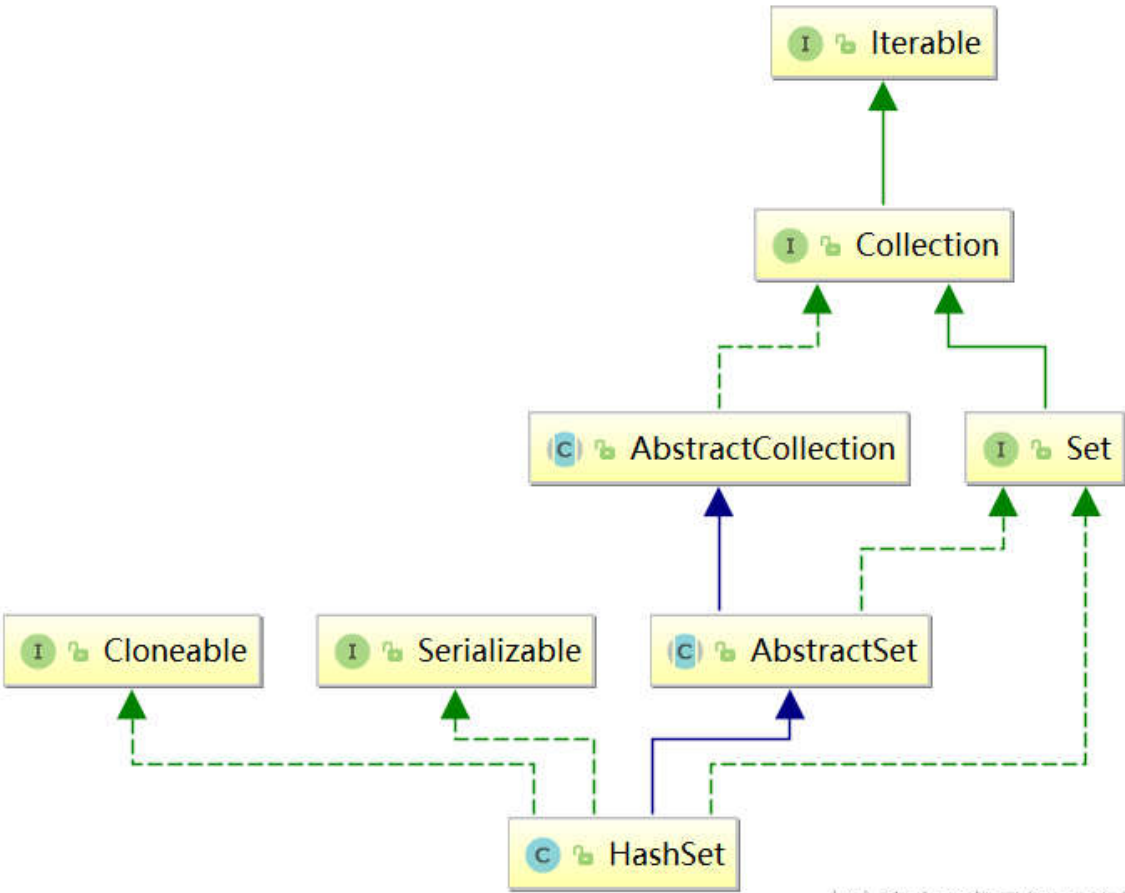


• `Vector`的类图结构:



牛客@我是祖国的花朵

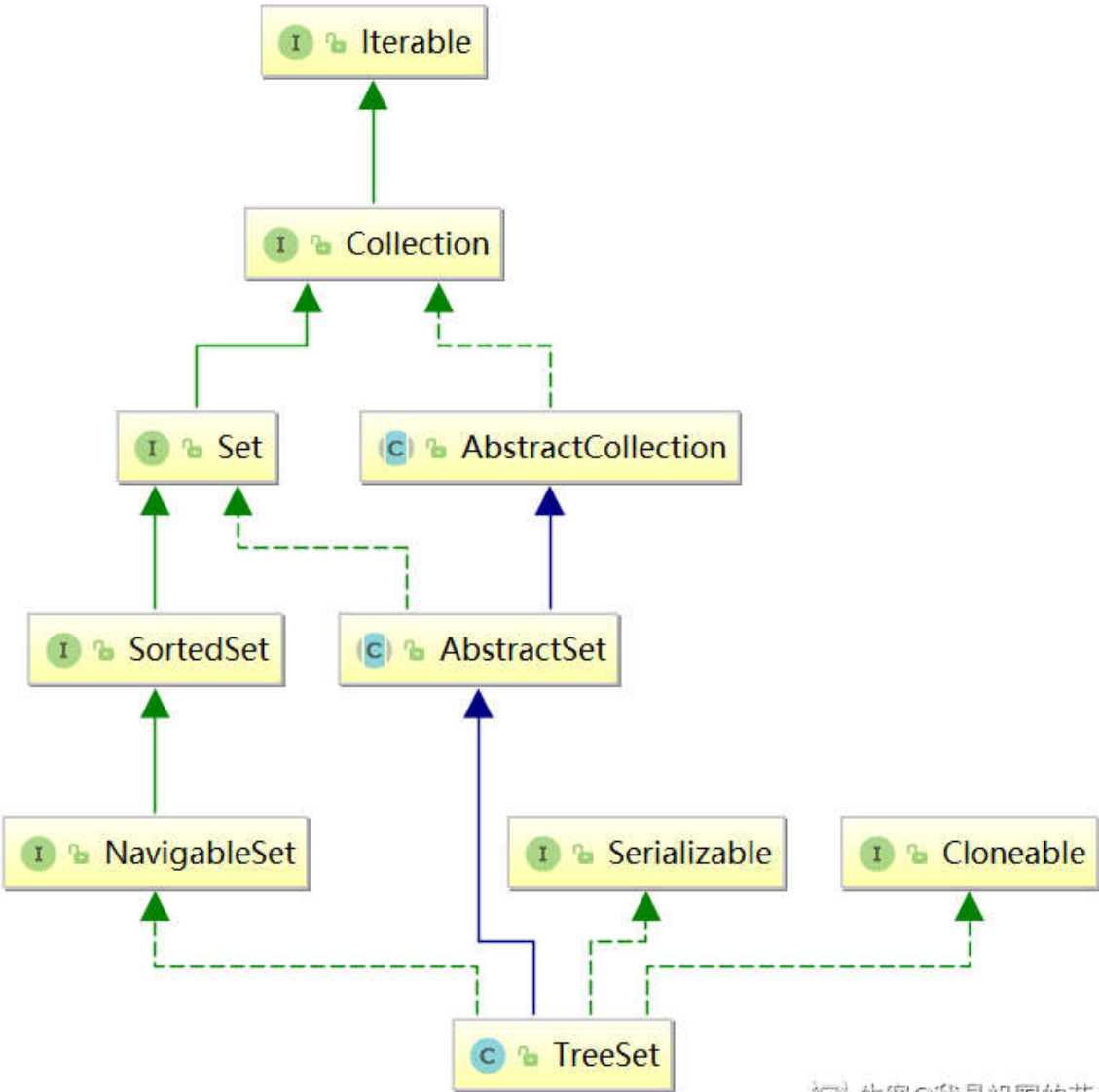
• HashSet的类图结构:



牛客@我是祖国的花朵

• TreeSet的类图结构:





牛客@我是祖国的花朵

讨论

评论



一只没有感情的鸽子

1#

打卡，期待更新

发表于 2019-11-25 17:10:30

赞(0) 回复(0)



我是祖国的花朵 N 作者

2#

既然，有同学希望我可以给出答案，而不是诱导大家去自行研究，OK，没问题。这里给出最后一道题目的解析：Collection和Collections的关系：Collection是一个顶层集合接口，其子接口包括List和Set；而Collections是一个集合工具类，可以操作集合，比如说排序，二分查找，拷贝集合，寻找最大最小值等。总而言之：带s的大都是工具类。

发表于 2019-11-27 22:49:09

赞(11) 回复(4)

从心出发(🌟🌟🌟)： 楼主你已经总结的很棒了，学习还是要自己思考，自己摸索

2019-11-28 11:47:45

赞(1) 回复(0)

我是祖国的花朵 N 作者 回复 从心出发(🌟🌟🌟)： 赞，加油

2019-11-28 12:53:29

赞(0) 回复(0)

LGSKOKO 回复 我是祖国的花朵 N (作者) : 楼主已经做的很棒了, 非常感谢。有些东西就是要自己去挖掘, 做IT的肯定都要有最基本的自学能力, 自己收集资料总结出来的印象会跟深刻。

2019-11-30 11:27:43赞(0) 回复(0)

Meloddy : 今天看了这节准备把这题总结一下的, 楼主已经写出来啦, 哈哈哈哈哈

2020-01-05 15:15:39赞(0) 回复(0)

请输入你的观点

回复

17



我是祖国的花朵 N (作者)3#

各位同学, 目前该篇文章中 " 集合 " 两字被编辑器认为是敏感字段, 自动替换成了\*\*\*符号。 预计下周修复该问题, 抱歉。 >\_<

发表于 2019-11-28 12:55:29赞(0) 回复(0)



夏洛克2019041410351994#

写得很棒, 作者加油! 不过希望更新速度快点, 最后还想知道全部更新完大概会到什么时候?

发表于 2019-12-03 21:25:57赞(0) 回复(2)

我是祖国的花朵 N (作者) : 感谢支持, 因为牛客专栏系统还不太完善, 简单说就是存在或多或少的bug, 一定程度影响了更新速度哈。我会在保证质量的前提下, 尽快更新。加油

2019-12-03 21:31:11赞(1) 回复(0)

夏洛克201904141035199 回复 我是祖国的花朵 N (作者) : 节省了不少查资料的时间, 感谢。

2019-12-03 21:37:28赞(0) 回复(0)

请输入你的观点

回复



刘畅2019042116068165#

打卡

发表于 2019-12-30 21:33:26赞(0) 回复(0)



牧水s N6#

打卡

发表于 2020-01-11 23:11:04赞(0) 回复(0)



柳杰2019050110494207#

叮

发表于 2020-01-15 18:26:14赞(0) 回复(0)



柳杰201905011049420

8#

叮

发表于 2020-01-15 18:26:14

赞(0) 回复(0)




柳杰201905011049420

9#

叮

发表于 2020-01-15 18:26:14

赞(0) 回复(0)



小源20190118151956 N

10#

打卡

发表于 2020-02-17 13:14:20

赞(0) 回复(0)



Myyyy`

11#

打卡，早就该来了👍

发表于 2020-02-21 07:52:06

赞(0) 回复(0)



亦成风

12#

打卡

发表于 2020-02-22 22:08:28

赞(0) 回复(0)




snailClimbZG

13#

打卡

发表于 2020-02-23 22:09:05

赞(0) 回复(0)



Gaido

14#

老师，Comparator是接口，我看例子里是new Comparator;接口的话不是不可以实例化的吗

发表于 2020-02-25 23:37:18

赞(0) 回复(4)

我是祖国的花朵 N (作者)： new一个接口的前提是必须实现接口中的抽象方法，其实就是相当于创建了一个一次性使用的实现类。比如new Runnable () 必须同时实现其run方法

2020-02-26 20:17:47

赞(0) 回复(0)

Gaido： 好的，谢谢老师

2020-02-27 14:35:36

赞(0) 回复(0)

爱生活的猿： 匿名内部类啊大哥，快要春招了，先搞定java基础再来看面试题，听我的，不然效果不大

2020-03-08 18:47:52

赞(0) 回复(0)

Gaido 回复 爱生活的猿： 好的，谢谢老哥

2020-03-08 18:58:01

赞(0) 回复(0)

请输入你的观点

回复

17



老宋啊啊啊

15#

第三天打卡，大写字母加了s通常都是工具类

发表于 2020-03-02 13:01:14

赞(0) 回复(0)



若木心

16#

叮

发表于 2020-03-10 00:43:56

赞(0) 回复(0)



sky爱吃青菜

17#

我的笔记：

(1) ArrayList初始容量为10，扩容是增加原来的一半，容量变为  $n+n/2$

(2) List.toArray()传入的是Integer这样的数组，而不是int

发表于 2020-06-20 21:51:01

赞(0) 回复(0)