**1. 常见的集合有哪些？**

答：**Map接口**和**Collection接口**是所有结合框架的父接口；

①Collection接口的子接口包括：Set接口和List接口；

②Map接口的实现类主要有：HashMap, TreeMap , Hashtable, ConcurrentHashMap以及Properties等；

③Set接口的实现类主要有：HashSet，TreeSet，LinkedHashSet等；

④List接口的实现类主要有：ArrayList，LinkedList，Stack, Vector等。

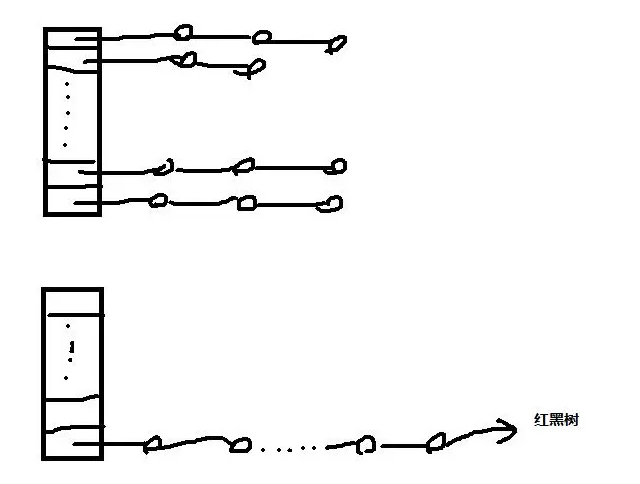
**2. HashMap和Hashtable的区别有哪些？**

答：①HashMap没有考虑同步，是线程不安全的；Hashtable使用了synchronized关键字，是线程安全的；

②前者允许null作为key；后者不允许null作为key。

**3. HashMap的底层实现知道吗？**

答：在Java8之前，其底层实现是数组+链表组成；Java8使用了**数组+链表+红黑树**实现。



**红黑树**：平衡二叉查找树

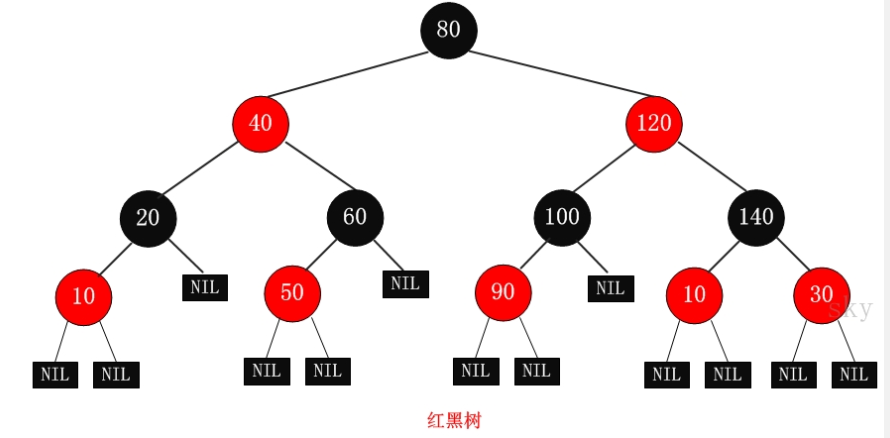
①每个节点或者是黑色，或者是红色；

②根节点必须是黑色；

③每个叶子节点都是黑色（注：这里的叶子节点是指为空的叶子节点）

④如果一个节点是红色的，那么它的子节点必须是黑色的；

⑤从一个节点到该节点的子孙节点的所有路径上包含相同数目的黑节点。



**4. ConcurrentHashMap和Hashtable的区别？**

答：ConcurrentHashMap结合HashMap和HashTable二者的优势。HashMap没有考虑同步，HashTable考虑了同步的问题。但是HashTable在每次同步执行时都要锁住整个结构。ConcurrentHashMap锁的方式是**稍微细粒度**的。CurrentHashMap将hash表分为16个桶（默认值），诸如get，put，remove等常用操作只锁当前需要用到的锁。

面试官：ConcurrentHashMap的具体实现知道吗？

答：①该类包含两个静态内部类**HashEntry和Segment**；前者用来封装映射表的键值对，后者用来充当锁的角色。

②Segment是一种**可重入的锁Reentrant Lock**，每个Segment守护一个HashEntry数组里的元素，当对HashEntry数组的数据进行修改时，必须首先获得对应锁的Segment锁。

**5.HashMap的长度为什么是2的幂次方？**

答：①通过将Key的hash值与length-1**进行&运算**，实现了当前Key的定位，2的幂次方可以减少冲突的次数，提高HashMap的查询效率；

②如果length为2的次幂则length-1转化为二进制必定是11111……的形式，在与h的二进制&操作效率会非常快，而且空间不浪费；如果length不是2的次幂，比如length为15，则length-1为14，对应的二进制为1110，在与h进行&操作，最后一位都为0，而0001，0011，0101，1001，1011，1101，0111这几个位置永远不能存放元素，空间浪费相当大，更糟的是这种情况中，数组可以使用的位置会比数组长度小很多，这意味着进一步增加了冲突的几率，减慢了查询的效率？这样就造成空间的浪费。

**6.List和Set的区别是什么？**

答：List元素是有序的，可以重复的；Set元素是无序的，不可以重复。

**7.List、Set和Map的初始容量和加载因子？**

答：①List: ArrayList的初始容量是10；加载因子为0.5；扩容增量：原容量的0.5倍+1；一次扩容后长度为16. Vector初始容量是10；加载因子是1；扩容增量：原容量的1倍，如Vector的容量为10，一次扩容后容量为20.

②Set HashSet初始容量是16，加载因子为0.75；扩容增量：原容量的1倍，如HashSet容量为16，一次扩容过后容量为32

③Map HashMap，初始容量为16，加载因子为0.75；扩容增量：原增量的1倍，如HashMap的容量为16，一次扩容后容量为32.

**8.Comparable接口和Comparator接口有什么区别**？

答：前者简单，但是如果需要重新定义比较类型时，需要修改源代码；

后者不需要修改源代码，自定义一个比较器，实现自定义的比较方法。

**9.Java集合的快速失败机制“fail-fast“**

答：时Java集合的一种**错误检测机制**，当多个线程对集合进行结构上的改变操作时，有可能会产生fail-fast机制。

**例如：**假设存在两个线程（线程1、线程2），线程1通过Iterator在遍历集合A中的元素，在某个时刻线程2修改了集合A的结构（是结构上的修改，而不是简单的修改集合元素的内容），那么这个时候程序就会抛出ConcurrentModificationException异常，从而产生fail-fast机制。

**原因：**迭代器在遍历时直接访问集合中的内容，并且在遍历过程中使用一个**modCount变量**。集合在被遍历期间如果内容发生变化，就会改变modCount的值。每当迭代器使用hashNext（）/next（）遍历下一个元素之前，都会检测modCount变量是否为exceptedmodCount值，是的话就返回遍历，否则抛出异常，终止遍历。

**解决方法**：①在遍历过程中，所有涉及到改变modCount值得地方全部加上synchronized。②使用CopyOnWriteArrayList来替换ArrayList.

**10.ArrayList和Vector的区别？**

答：这两个类都实现了List接口（List接口继承了Collection接口），他们都是有序集合，即**存储在着两个集合中的元素位置都是有顺序的，相当于一种动态的数组**。我们可以按位置索引来取出某个元素，并且其中的数据时允许重复的，这是与HashSet之类的集合的最大不同。HashSet之类的集合不可以按索引号去检索其中的元素，也不允许有重复的元素。

ArrayList和Vector的区别主要包括两个方面：

①同步性。  **Vector是线程安全的，也就是说它的方法之间是线程同步的（加了synchronized关键字），而ArrayList是线程不安全的，它的方法之间是线程不同步的。**如果只有一个线程会访问到集合，那最好是使用ArrayList，因为它不考虑线程安全问题，所以效率会更高一些；如果有多个线程会访问到集合，那最好是使用Vector，因为不需要我们自己去考虑和编写线程安全的代码。

②数据增长。 ArrayList和Vector都有一个初始的容量大小，当存储进它们里面的元素的数目超过了容量时，就需要增加ArrayList和Vector的存储空间，每次要增加存储空间时，不是只增加一个存储单元，而是增加多个存储单元。Vector在数据满时（加载因子为1）增长为原来的1倍，而ArrayList在数据量达到容量的一半时（加载因子为0.5）增长为原容量的0.5倍+1。

**面试官：那ArrrayList和LinkedList的区别呢？**

答：①**LinkedList实现了List和Deque接口，一般称为双向链表；**

②LinkedList在插入和删除数据时效率更高，ArrayList在查找某个index的数据时效率更高；

③LinkedList比ArrayList需要更多的内存。

**面试官：Array和ArrayList有什么区别？什么时候用Array而不是ArrayList?**

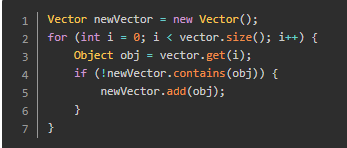
①Array可以包含基本类型和对象类型，但ArrayList只能包含对象类型；

②Array是固定大小的，ArrayList的大小是动态改变的；

③ArrayList提供了更多的方法和特性，比如：addAll() removeAll() iterator()。。。

对于基本类型数据，集合使用自动箱来减少编码工作量。但是，当处理固定大小的基本数据类型的时候，这种方式相对比较慢。

**11.如何去掉一个Vector集合中重复的元素？**



**12.如何权衡是使用无序的数组还是有序的数组？**

答：有序数组最大的好处在于查找的时间复杂度是O(log n)而无序数组是O(n)。有序数组的缺点是插入操作的时间复杂度是O（n），因为值大的元素需要往后移动来给新元素腾位置。相反，无序数组的插入时间复杂度是常量O(1)