面试题——容器

1. java 容器都有哪些？

答：List、Set、Map

2. Collection和Collections有什么区别？

答：Collection是集合类的顶级接口，其派生了两个子接口 Set 和 List。

　　Collections则是集合类的一个工具类/帮助类，其中提供了一系列静态方法，用于对集合中元素进行排序、搜索以及线程安全等各种操作。

总的来说：Collection是一个接口，而Collections是个类。

3. List、Set、Map 之间的区别是什么？

答：List：可以允许重复对象；可以插入多个null元素；是一个有序容器

Set：不允许重复对象；只允许一个null元素；是一个无序容器

Map：Map不是Collection的子接口或实现类。Map是一个接口，Map 的每个Entry都特有两个对象，也就是一个键一个值，Map可能会持有相同的值对象但键对象必须是唯一的

Map里可以拥有随意个null值但最多只能有一个null键。

4. HashMap 和 Hashtable 有什么区别？

答：存储：HashMap允许key 和 value 为null，而Hashtable不允许

线程安全：Hashtable是线程安全的，而HashMap是非线程安全

HashMap是非synchronized，而Hashtable是synchronized

5. 如何决定使用 HashMap 还是 TreeMap？

答：对于在Map中插入、删除、定位一个元素这类操作，HashMap 是最好的选择，因为相对而言HashMap的插入会更快，但如果你要对一个key集合进行有序的遍历，那TreeMap是更好的选择。

6. 说一下 HashMap 的实现原理：

答：HashMap是基于hashing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。

7. 当两个对象的hashcode相同会发生什么？

答：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。

8. 如果两个键的hashcode相同，你如何获取值对象？

答：找到bucket位置之后，会调用keys.equals()方法去找到链表中正确的节点，最终找到要找的值对象。

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

答：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。

10. 你了解重新调整HashMap大小存在什么问题吗？

答：可能产生条件竞争(race condition)，为什么这么奇怪，要在多线程的环境下使用HashMap呢？

11. 我们可以使用ConcurrentHashMap来代替Hashtable吗？

答：我们知道Hashtable是synchronized的，但是ConcurrentHashMap同步性能更好，因为它仅仅根据同步级别对map的一部分进行上锁。ConcurrentHashMap当然可以代替HashTable，但是HashTable提供更强的线程安全性。

12. 说一下HashSet的实现原理？

答：①是基于HashMap实现的，默认构造函数是构建一个初始容量为16，负载因子为0.75 的HashMap。封装了一个HashMap对象来存储所有的集合元素，所有放入 HashSet 中的集合元素实际上由HashMap的key来保存，而HashMap的value则存储了一个PRESENT，它是一个静态的Object对象。

②当我们试图把某个类的对象当成 HashMap的 key，或试图将这个类的对象放入HashSet 中保存时，重写该类的equals(Object obj)方法和hashCode()方法很重要，而且这两个方法的返回值必须保持一致：当该类的两个的hashCode()返回值相同时，它们通过 equals() 方法比较也应该返回true。通常来说，所有参与计算hashCode()返回值的关键属性，都应该用于作为equals()比较的标准。

③HashSet的其他操作都是基于HashMap的。

13. ArrayList 和 LinkedList 的区别是什么？

答：ArrayList 与 LinkedList都实现了List接口

ArrayList 是线性表，底层是使用数组实现的，它在尾端插入和访问数据时效率较高

LinkedList 是双向链表，它在中间插入或者插入时效率较高，在访问数据时效率较低

14. 如何实现数组和List之间的转换？

答：List转数组：toArray(arraylist.size())方法

数组转List：Arrays的asList(a)方法

15. ArrayList 和 Vector 的区别是什么？

答：（1）同步性：Vector是线程安全的，用synchronized实现线程安全，而ArrayList是线程不安全的，如果只有一个线程会访问到集合，那最好使用ArrayList，因为它不考虑线程安全，效率会高些；如果有多个线程会访问到集合，那最好是使用Vector，因为不需要我们再去考虑和编写线程安全的代码。

（2）数据容量增长：二者都有一个初始容量大小，采用线性连续存储空间，当存储的元素的个数超过了容量时，就需要增加二者的存储空间，Vector增长原来的一倍，ArrayList增加原来的0.5倍。

16. Array 和 ArrayList 有何区别？

答：ArrayList是一种“会自动扩增容量的Array”。 Array（[]）：最高效；但是其容量固定且无法动态改变。ArrayList：容量可动态增长但牺牲效率；

17. 在 Queue 中 poll()和 remove()有什么区别？

答：remove()和poll()方法都是从队列中删除第一个元素。如果队列元素为空，调用remove() 的行为与Collection接口的版本相似会抛出异常，但是新的poll()方法在用空集合调用时只是返回 null。因此新的方法更适合容易出现异常条件的情况。

18. 哪些集合类是线程安全的？

答：Vector：就比Arraylist多了个同步化机制（线程安全）。

Hashtable：就比Hashmap多了个线程安全。

ConcurrentHashMap: 是一种高效但是线程安全的集合。

Stack：栈，也是线程安全的，继承于Vector。

19. 迭代器 Iterator 是什么？

答：迭代器是一个“可遍历STL容器内全部或者部分元素”的对象

一个迭代器指出容器中的一个特定位置，具有遍历复杂数据结构的能力

20. Iterator 怎么使用？有什么特点？

答：（1）怎么使用：boolean hasNext() 判断迭代器中是否还有下一个元素，有则返回true

Object next()返回迭代器中下一个元素

Void remove()删除集合里上一个next方法调用的时候返回的对象元素

void forEachRemaining(Consumer action)使用Lambdda表达式的形式输出Iterator中所以的元素。注意该方法其实是间接调用next()方法进行遍历，

所以再次是next()方法的时候Iterator中的对象已经被遍历完了。

（2）特点：Iterator遍历集合元素的过程中不允许线程对集合元素进行修改，否则会抛出ConcurrentModificationException的异常。

Iterator遍历集合元素的过程中可以通过remove方法来移除集合中的元素。

Iterator必须依附某个Collection对象而存在，Iterator本身不具有装载数据对象的功能。

Iterator.remove方法删除的是上一次Iterator.next()方法返回的对象。

强调以下next()方法，该方法通过游标指向的形式返回Iterator下一个元素。

21. Iterator 和 ListIterator 有什么区别？

答：1. ListIterator有add()方法，可以向List中添加对象，而Iterator不能

2. ListIterator和Iterator都有hasNext()和next()方法，可以实现顺序向后遍历，但是ListIterator有hasPrevious()和previous()方法，可以实现逆向（顺序向前）遍历。Iterator就不可以。

3. ListIterator可以定位当前的索引位置，nextIndex()和previousIndex()可以实现。Iterator没有此功能。

4. 都可实现删除对象，但是ListIterator可以实现对象的修改，set()方法可以实现。Iierator仅能遍历，不能修改。

22. 怎么确保一个集合不能被修改？

答：使用final关键字