容器：

18.Java容器有哪些

分为Collection和Map两大类

Collection分为Queue,List,Set //都是接口，例如List<Integer> xx=new ArrayList<Integer>，子类必须实现接口所有功能才能实例化，不然看作抽象类

List：想想arrayList，特点就是有序，允许重复

下有ArrayList, LinkedList, Vector

Stack继承了Vector

Set: 集合，默认无序，不允许重复

下有HashSet, TreeSet（他是排序的，不是先进去Index就是0，而是561234这样插进去，会自动排成123456）

Queue: 单进单出，只能操作两端

常用实现类是LinkedList, Priority Queue

Map:特点就是Key,Value

常用实现类是HashMap,HashTable(别用，hashMap父类是AbstractMap,而HashTable父类是Dictionary,Dictionary已经被废弃了，所以它的子类也被废弃了),ConcurrentHashMap作为代替

### 1. 请说一下Java容器集合的分类，各自的继承结构

* Java中的容器集合分为两大阵营，一个是Collection，一个是Map
* Collection下分为Set，List，Queue
* Set的常用实现类有HashSet，TreeSet等
* List的常用实现类有ArrayList，LinkedList等
* Queue的常用实现类有LinkedList，ArrayBlockingQueue等
* Map下没有进一步分类，它的常用实现类有HashMap，ConcurrentHashMap等

2. 请谈一谈Java集合中的fail-fast和fail-safe机制

fail-fast是一种错误检测机制，Java在适合单线程使用的集合容器中很好地实现了fail-fast机制，举一个简单的例子：在多线程并发环境下，A线程在通过迭代器遍历一个ArrayList集合，B线程同时对该集合进行增删元素操作，这个时候线程A就会抛出并发修改异常，中断正常执行的逻辑。

java.util包下的集合类都是快速失败的，不能在多线程下发生并发修改（迭代过程中被修改）

而fail-safe机制更像是一种对fail-fast机制的补充，它被广泛地实现在各种并发容器集合中。回头看上面的例子，如果线程A遍历的不是一个ArrayList，而是一个CopyOnWriteArrayList，则符合fail-safe机制，线程B可以同时对该集合的元素进行增删操作，线程A不会抛出任何异常。

场景：java.util.concurrent包下的容器都是安全失败，可以在多线程下并发使用，并发修改。

一：快速失败（fail—fast）

         集合对象有一个modCount变量，如果遍历过程中对集合对象的内容进行了修改（增加、删除、修改），modCount自增，每当迭代器使用hashNext()/next()遍历下一个元素之前（每次获取下一个元素的时候），都会检测modCount变量是否为expectedmodCount值，如果不相等（也就是另外一个进程也改变了这个集合对象）就抛出异常。

二：安全失败（fail—safe）

      采用安全失败机制的集合容器，在遍历时不是直接在集合内容上访问的，而是先复制原有集合内容，在拷贝的集合上进行遍历。

      原理： 当使用迭代器遍历集合时，会基于原数组拷贝出一个新的数组（ArrayList的底层是数组），后续的遍历行为在新数组上进行。所以线程B同时进行增删操作不会影响到线程A的遍历行为。

### 3. 如何一边遍历一边删除Collection中的元素？

使用集合迭代器自身的remove方法进行删除

Iterator<Integer> it = list.iterator();

while(it.hasNext()){

\*// do something\*

it.remove();

}

### 4. 谈谈ArrayList和LinkedList的区别

本质的区别来源于两者的底层实现：ArrayList的底层是数组，LinkedList的底层是双向链表。

数组拥有O(1)的查询效率，可以通过下标直接定位元素；链表在查询元素的时候只能通过遍历的方式查询，是O(n)

数组增删元素的效率比较低，通常要伴随拷贝数组的操作；链表增删元素的效率很高，只需要调整对应位置的指针即可。

以上是数组和链表的通俗对比，在日常的使用中，两者都能很好地在自己的适用场景发挥作用。

比如说我们常常用ArrayList代替数组，因为封装了许多易用的api，而且它内部实现了自动扩容机制，由于它内部维护了一个当前容量的指针size，直接往ArrayList中添加元素的时间复杂度是O(1)的，使用非常方便。

而LinkedList常常被用作Queue队列的实现类，由于底层是双向链表，能够轻松地提供先入先出的操作。

19. Collection 和 Collections 有什么区别？

Collection 是一个集合接口，它提供了对集合对象进行基本操作的通用接口方法，所有集合都是它的子类，比如 List、Set 等。

Collections 是一个包装类，包含了很多静态方法，不能被实例化，就像一个工具类，比如提供的排序方法： Collections. sort(list)。

20.List、Set、Map 之间的区别是什么？

****