java ??????

主要介绍　　JAVA 基本数据结构

- ArrayList 与 LinkList异同__

ArrayList ?LinkList??

ArrayList和LinkedList都实现了List接口，他们有以下的不同点：
ArrayList是基于索引的数据接口，它的底层是数组。它可以以O(1)时间复杂度对元素进行随机访问。与此对应，LinkedList是以元素列表的形式存储它的数据，每一个元素都和它的前一个和后一个元素链接在一起，在这种情况下，查找某个元素的时间复杂度是O(n)。
相对于ArrayList，LinkedList的插入，添加，删除操作速度更快，因为当元素被添加到集合任意位置的时候，不需要像数组那样重新计算大小或者是更新索引。
LinkedList比ArrayList更占内存，因为LinkedList为每一个节点存储了两个引用，一个指向前一个元素，一个指向下一个元素。
也可以参考ArrayList vs. LinkedList。

1) 因为 Array 是基于索引 (index) 的数据结构，它使用索引在数组中搜索和读取数据是很快的。 Array 获取数据的时间复杂度是 O(1), 但是要删除数据却是开销很大的，因为这需要重排数组中的所有数据。

2) 相对于 ArrayList ， LinkedList 插入是更快的。因为 LinkedList 不像 ArrayList 一样，不需要改变数组的大小，也不需要在数组装满的时候要将所有的数据重新装入一个新的数组，这是 ArrayList 最坏的一种情况，时间复杂度是 O(n) ，而 LinkedList 中插入或删除的时间复杂度仅为 O(1) 。 ArrayList 在插入数据时还需要更新索引（除了插入数组的尾部）。

3) 类似于插入数据，删除数据时， LinkedList 也优于 ArrayList 。

4) LinkedList 需要更多的内存，因为 ArrayList 的每个索引的位置是实际的数据，而 LinkedList 中的每个节点中存储的是实际的数据和前后节点的位置 (一个 LinkedList 实例存储了两个值： Node<E> first 和 Node<E> last 分别表示链表的其实节点和尾节点，每个 Node 实例存储了三个值： E item,Node next,Node pre) 。

什么场景下更适宜使用 LinkedList，而不用ArrayList

1) 你的应用不会随机访问数据 。因为如果你需要LinkedList中的第n个元素的时候，你需要从第一个元素顺序数到第n个数据，然后读取数据。

2) 你的应用更多的插入和删除元素，更少的读取数据 。因为插入和删除元素不涉及重排数据，所以它要比ArrayList要快。

○ </>

3) 类似于插入数据，删除数据时， LinkedList 也优于 ArrayList 。

4) LinkedList 需要更多的内存，因为 ArrayList 的每个索引的位置是实际的数据，而 LinkedList 中的每个节点中存储的是实际的数据和前后节点的位置 (一个 LinkedList 实例存储了两个值： Node<E> first 和 Node<E> last 分别表示链表的其实节点和尾节点，每个 Node 实例存储了三个值： E item,Node next,Node pre) 。

什么场景下更适宜使用 LinkedList，而不用ArrayList

1) 你的应用不会随机访问数据 。因为如果你需要LinkedList中的第n个元素的时候，你需要从第一个元素顺序数到第n个数据，然后读取数据。