1.说说有哪些常见的集合框架?

- 推荐阅读：二哥的 Java 进阶之路：Java 集合框架 ↗
- 推荐阅读：阻塞队列 BlockingQueue ↗。

Java 集合框架可以分为两条大的支线：

①、Collection，主要由 List、Set、Queue 组成：

- List 代表有序、可重复的集合，典型代表就是封装了动态数组的 ArrayList ↗ 和封装了链表的 LinkedList ↗；
- Set 代表无序、不可重复的集合，典型代表就是 HashSet 和 TreeSet；
- Queue 代表队列，典型代表就是双端队列 ArrayDeque ↗，以及优先级队列 PriorityQueue ↗。

②、Map，代表键值对的集合，典型代表就是 HashMap ↗。

2.ArrayList 和 LinkedList 有什么区别?

推荐阅读：二哥的 Java 进阶之路：ArrayList 和 LinkedList ↗

ArrayList 是基于数组实现的，LinkedList 是基于链表实现的。

用途有什么不同?

多数情况下，ArrayList 更利于查找，LinkedList 更利于增删

使用场景有什么不同?

ArrayList 适用于：

- 随机访问频繁：需要频繁通过索引访问元素的场景。
- 读取操作远多于写入操作：如存储不经常改变的列表。
- 末尾添加元素：需要频繁在列表末尾添加元素的场景。

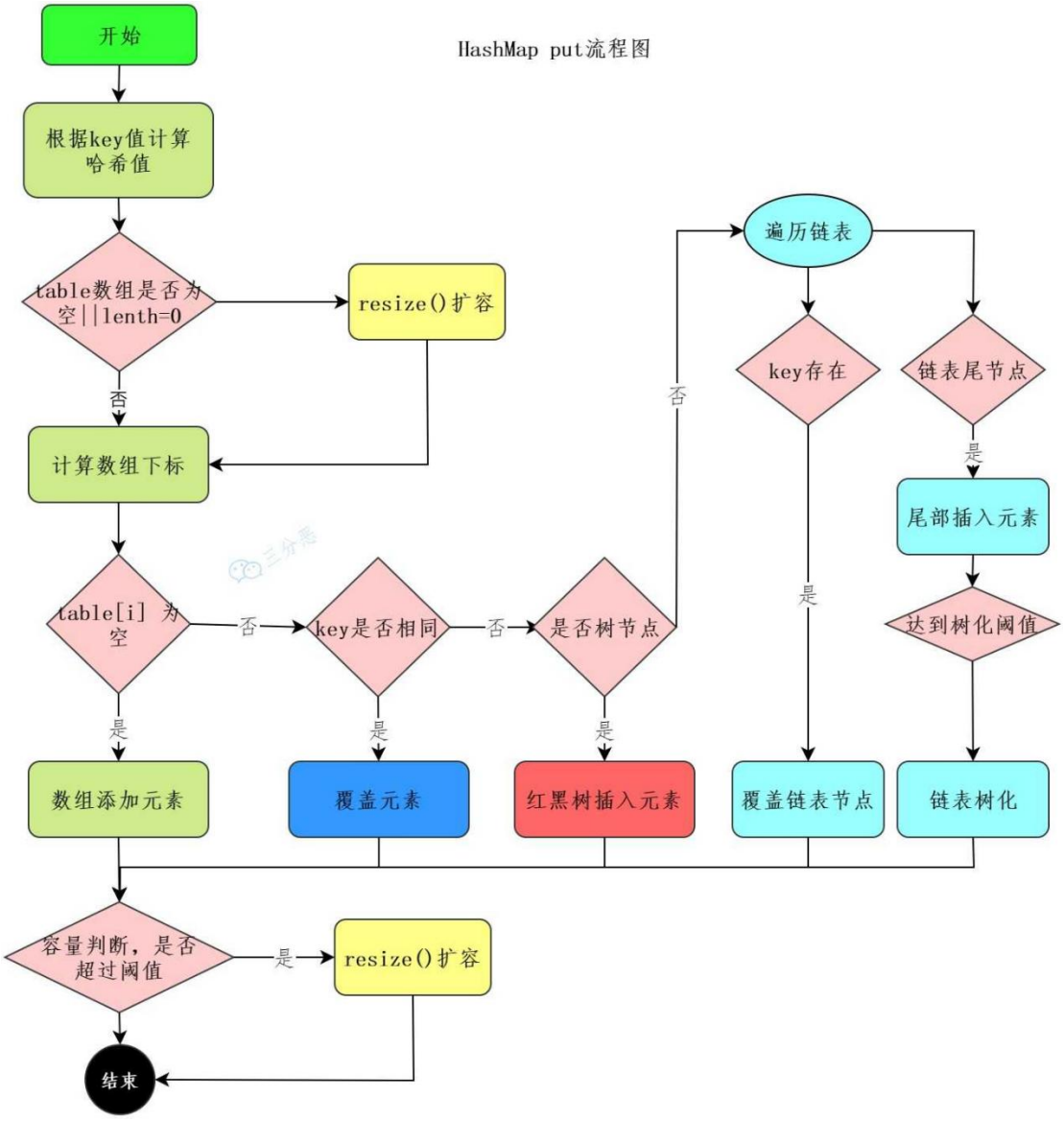
LinkedList 适用于：

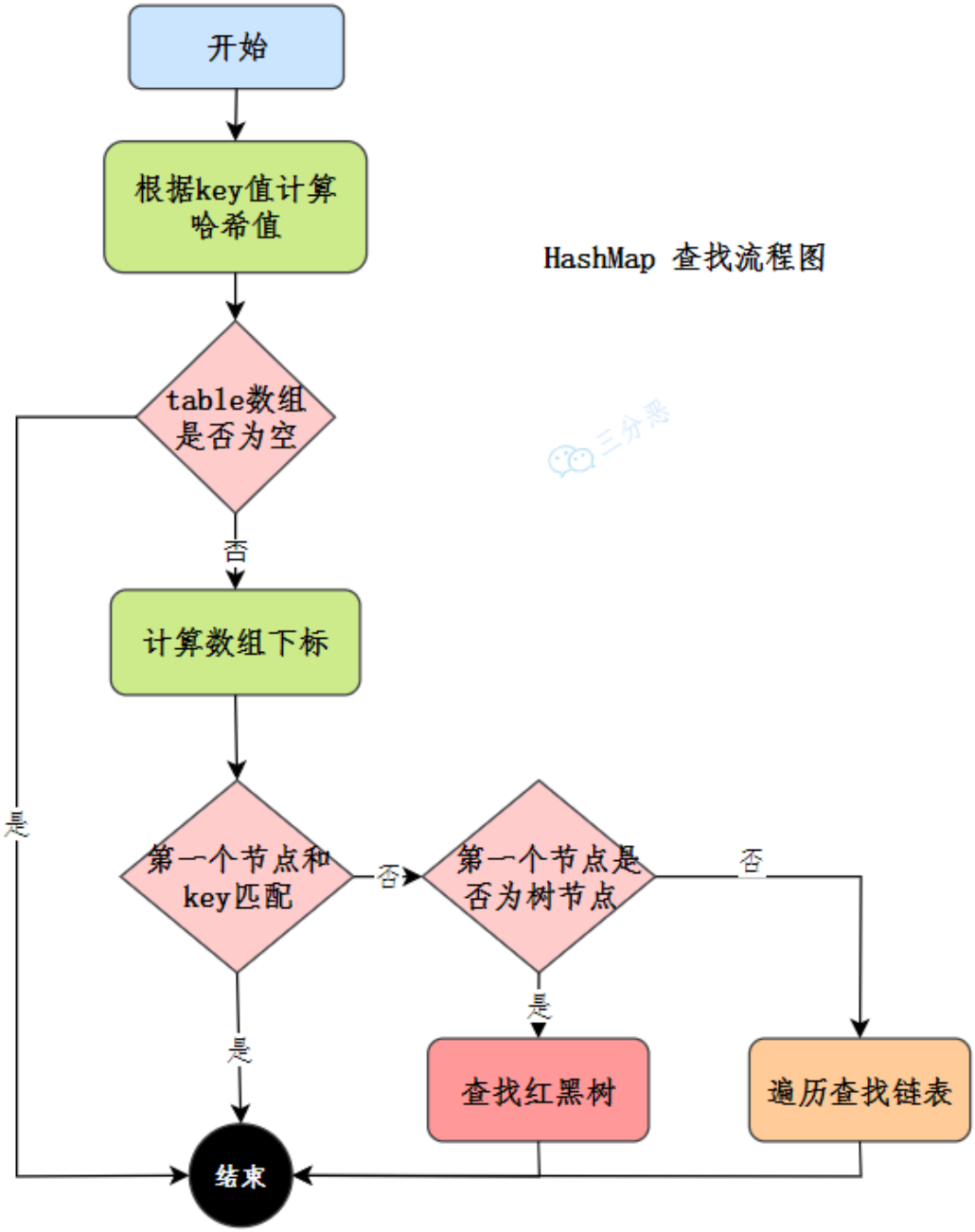
- 频繁插入和删除：在列表中间频繁插入和删除元素的场景。
- 不需要快速随机访问：顺序访问多于随机访问的场景。
- 队列和栈：由于其双向链表的特性，LinkedList 可以高效地实现队列（FIFO）和栈（LIFO）。

8.能说一下 HashMap 的底层数据结构吗?

推荐阅读：二哥的 Java 进阶之路：详解 HashMap ↗

JDK 8 中 HashMap 的数据结构是 `数组` + `链表` + `红黑树` 。





HashMap 查找流程图

18.解决哈希冲突有哪些方法呢?

解决哈希冲突的方法我知道的有 3 种:

①、再哈希法

准备两套哈希算法，当发生哈希冲突的时候，使用另外一种哈希算法，直到找到空槽为止。对哈希算法的设计要求比较高。

②、开放地址法

遇到哈希冲突的时候，就去寻找下一个空的槽。有 3 种方法:

- 线性探测：从冲突的位置开始，依次往后找，直到找到空槽。
- 二次探测：从冲突的位置 x 开始，第一次增加 1^2 个位置，第二次增加 2^2，直到找到空槽。
- 双重哈希：和再哈希法类似，准备多个哈希函数，发生冲突的时候，使用另外一个哈希函数。

22.JDK 8 对 HashMap 主要做了哪些优化呢? 为什么?

相比较 JDK 7，JDK 8 的 HashMap 主要做了四点优化:

①、底层数据结构由数组 + 链表改成了数组 + 链表或红黑树的结构。

原因：如果多个键映射到了同一个哈希值，链表会变得很长，在最坏的情况下，当所有的键都映射到同一个桶中时，性能会退化到 O(n)，而红黑树的时间复杂度是 O(logn)。

②、链表的插入方式由头插法改为了尾插法。

原因：头插法虽然简单快捷，但扩容后容易改变原来链表的顺序。

③、扩容的时机由插入时判断改为插入后判断。

原因：可以避免在每次插入时都进行不必要的扩容检查，因为有可能插入后仍然不需要扩容。

④、优化了哈希算法。

JDK 7 进行了多次移位和异或操作来计算元素的哈希值。

③、拉链法

也就是所谓的链地址法，当发生哈希冲突的时候，使用链表将冲突的元素串起来。HashMap 采用的正是拉链法。

27.讲讲 LinkedHashMap 怎么实现有序的?

LinkedHashMap 维护了一个双向链表，有头尾节点，同时 LinkedHashMap 节点 Entry 内部除了继承 HashMap 的 Node 属性，还有 before 和 after 用于标识前置节点和后置节点。

1.得到的新的数组索引和老数组索引只有最高位区别，更快地得到新索引

2.rehash时的取余操作，hash % length == hash & (length - 1)这个关系只有在length等于二的幂次方时成立，位运算能比%高效得多

https://blog.csdn.net/weixin_44273302/article/details/113733422