**hashMap扩容**

什么时候扩容：当向容器添加元素的时候，会判断当前容器的元素个数，如果大于等于阈值---即当前数组的长度乘以加载因子的值的时候，就要自动扩容啦。

扩容(resize)：当HashMap对象内部的数组无法装载更多的元素时，对象就需要扩大数组的长度，以便能装入更多的元素。Java里的数组是无法自动扩容的，所以使用一个新的数组代替已有的容量小的数组。

resize就是使用一个容量更大的数组来代替已有的容量小的数组，transfer()方法将原有Entry数组的元素拷贝到新的Entry数组里。如果发生了hash冲突，使用单链表的头插入方式，同一位置上新元素总会被放在链表的头部位置，这样先放在一个索引上的元素终会被放到Entry链的尾部。在旧数组中同一条Entry链上的元素，通过重新计算索引位置后，有可能被放到了新数组的不同位置上。

rehash使用的是2次幂的扩展(指长度扩为原来2倍)，经过rehash之后，元素的位置要么是在原位置，要么是在原位置再移动2次幂的位置

Jdk1.7 hash(){return key % table.length;}

JDK1.8的优化

　　　　　　　　　　　　　　　　// 原索引

51 if ((e.hash & oldCap) == 0) {

52 if (loTail == null)

53 loHead = e;

54 else

55 loTail.next = e;

56 loTail = e;

57 }

58 // 原索引+oldCap

59 else {

60 if (hiTail == null)

61 hiHead = e;

62 else

63 hiTail.next = e;

64 hiTail = e;

65 }

JDK1.7中rehash的时候，旧链表迁移新链表的时候，如果在新表的数组索引位置相同，则链表元素会倒置，JDK1.8不会