**2.2 归并排序**

**2.2.1 原地归并的抽象方法**

归并，将两个有序的数组归并成一个更大的有序数组。

要将一个数组排序，可以先（递归地）将它分成两半分别排序，然后将结果归并起来。

归并能够保证将任意长度为N的数组排序所需时间和NlogN成正比，缺点是需要的额外空间和N成正比。

实现归并的一种直截了当的办法是将两个不同的有序数组归并到第三个数组中。

将两个输入数组中的元素一个个从小到大放入第三个数组中。

编程：原地归并的抽象方法

**2.2.2 自顶向下的归并排序**

编程：自顶向下的归并排序

对于长度为N的任意数组，自顶向下的归并排序需要1/2NlgN至NlgN次比较。

归并排序所需的时间和NlgN成正比。

对小规模数组使用插入排序。

使用插入排序处理小规模的子数组（比如长度小于15）一般可以将归并排序的运行时间缩短10%-15%。

添加一个判断条件，如果a[mid]小于等于a[mid+1]，我们就认为数组已经是有序的并跳过merge方法。

三个改进归并排序的方法：1，对小规模数组使用插入排序；2，测试数组是否已经有序；3，不将元素复制到辅助数组

**2.2.3 自底向上的归并排序**

递归实现的归并排序是算法设计中分治思想的典型应用。我们将大问题分割成小问题分别解决，然后用所有小问题的答案解决大问题。

实现归并排序的另一种方法是先归并那些微型数组，然后再成对归并得到的子数组，如此这般，直到我们将整个数组归并在一起。

首先我们进行的是两两归并，然后四四归并，八八归并……

编程：自底向上的归并排序

自底向上的归并排序比较适合用链表组织的数据。

**2.2.4 排序算法的复杂度**

没有任何基于比较的算法能够保证使用少于lg(N!)-NlgN次比较将长度为N的数组排序。