1. 合并搜索

初始版本由于访问的随机性，有可能导致所建立的树的高度过高

增强版本（书上所说的最好的目前版本）利用增加的权值（记录的节点数）在合并时进行选择性合并（记录高度会有什么不同么）

路径压缩算法：在检查节点的同时将他们直接连接到根节点

1. 归并排序

归并优化四部曲：

使用插入排序处理小规模的数组（例如长度小于15，小规模的排序继续调用递归函数比这个慢？）（插入排序比选择排序快，恩，书上的数据是会快一点）

通过增加判断条件避免在merge中对已经有序的数据进行多余的操作

通过在外部的数组申请，避免了在递归函数内部进行数组申请的麻烦

1. 堆排序

堆排序在速度方面显得有些捉急，比较的元素不相邻导致高速缓存的命中率很低，优点是在空间的利用上是很好的

下沉改进：

不是将最后一个元素和第一个元素进行交换然后进行三元素比较进行下沉，而是用一个额外存储空间暂时进行记录，通过对二叉树的两个元素进行直接比较最后空出位置放入最后一个元素

1. 快速排序

改进：

切换到插入排序

3取样切分，哨兵去除边界测试

对于拥有大量重复元素的数据，使用三向切分

1. 红黑树用一个附加变量代替二三树的二三节点
2. 散列表的优化