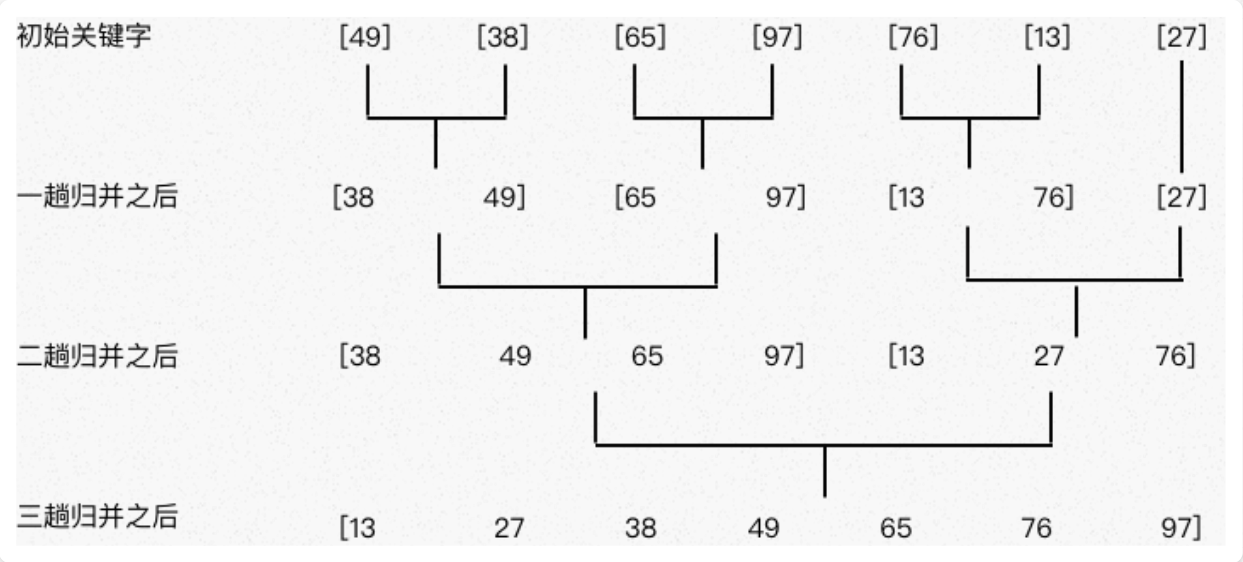
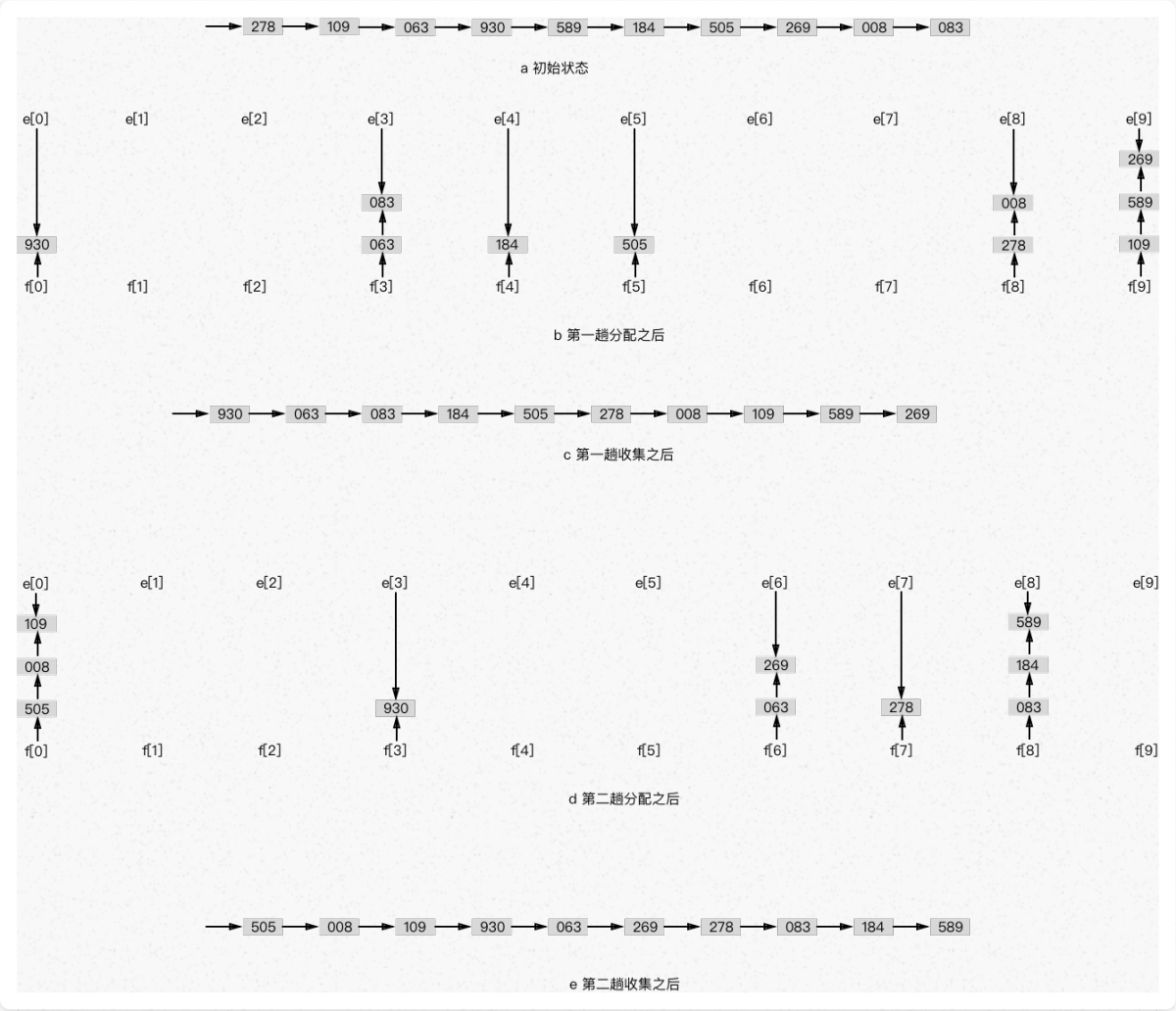
排序算法

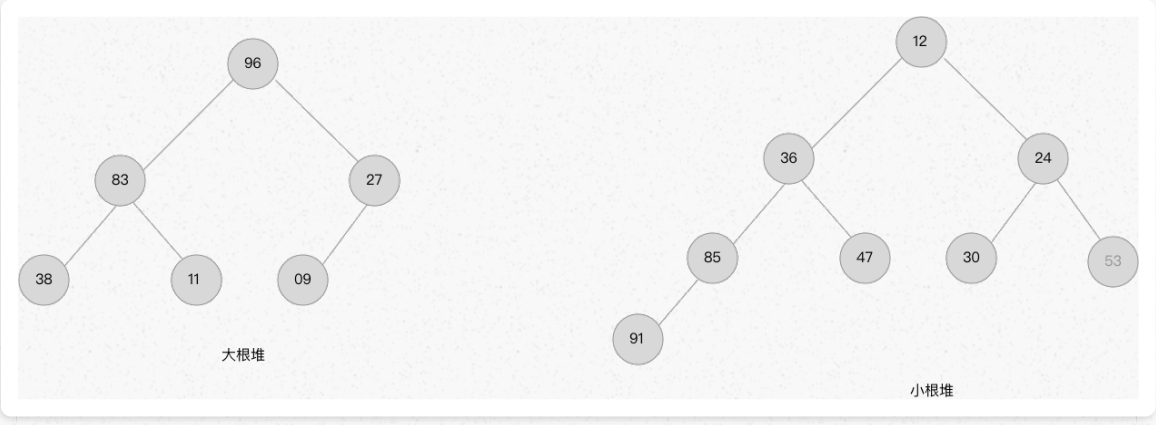
- 归并排序：将两个或两个以上的`有序`表合成一个有序表。
  - 算法思想：假设初始序列含有n个记录，则可以看成是n个`有序`的子序列，每个子序列的长度为1，然后两两归并，得到⌈n/2⌉个长度为2或1的有序子序列；再两两归并，....，如此重复，直到得到哟歌长度为n的有序序列为止。



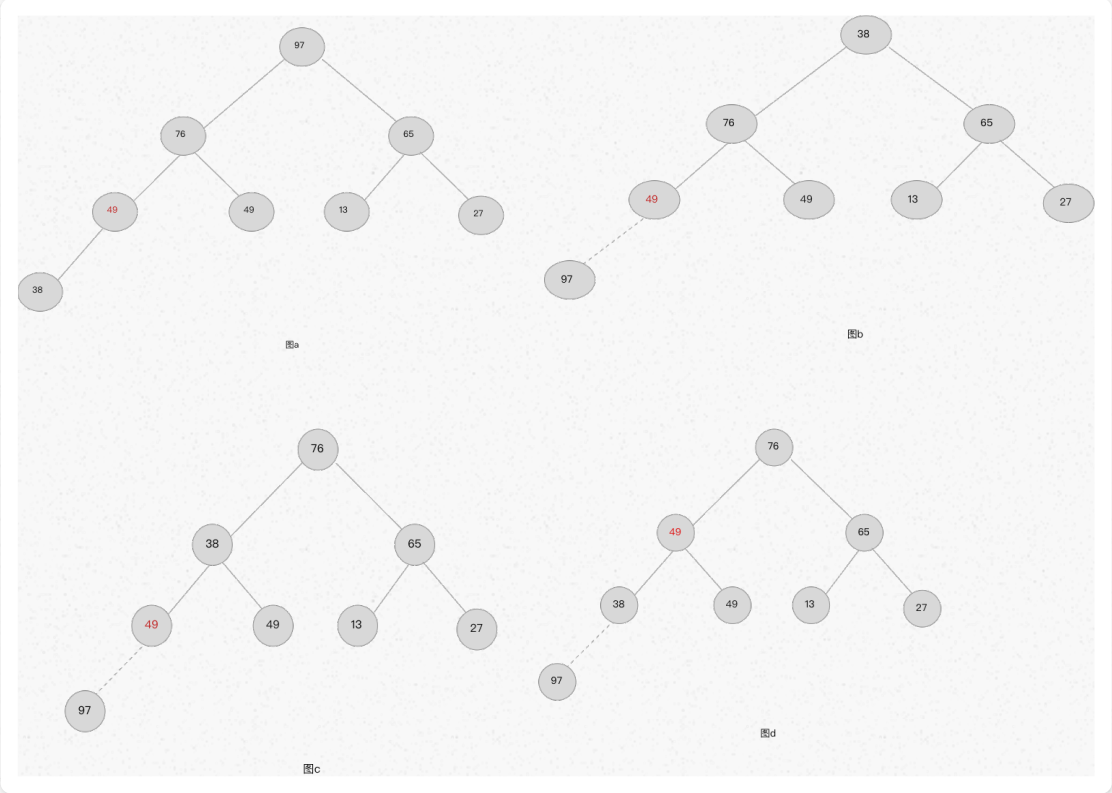
- 基数排序：根据关键字中各位的值，通过对待排序纪录进行若干趟“分配”与“收集”来实现排序，是一种借助于多关键字排序的思想对单关键字排序的方法。
  - 算法思想：假设记录的逻辑关键字由d个“关键字”组成，每个关键字可能取rd个值。只要从最低数位关键字起，按关键字的不同值将序列中纪录“分配”到rd个队列中之后再“收集”之，如此重复d次完成排序。其中“基”指的是rd的取值范围。例如：若关键字是数值，且其值都在0≤K≤999范围内，则可把每一个十进制数字看成一个关键字，既可以认为K由3个关键字（K0,K1,K2)组成，其中K0是百位数，K1是十位数，K2是个位数。



- 堆排序：堆排序是一种树形选择排序，在排序过程中，将待排序的纪录的纪录r[1..n]看成一棵完全二叉树的顺序储存结构，利用完全二叉树中双亲节点和孩子节点之间的内在关系，在当前无序的序列中选择关键字最大（或最小）的纪录。
  - 堆定义：n个元素的序列{k1,k2,...,kn}称之为堆，当且仅当满足一下条件
    - ki>=k2i且ki<=k2i+1 或 ki>=k2i且ki<=k2i+1
    - 堆实质上是满足如下性质的完全二叉树：树中所有非终端节点的值均不大于（或均不小于）其左、右孩子节点的值。



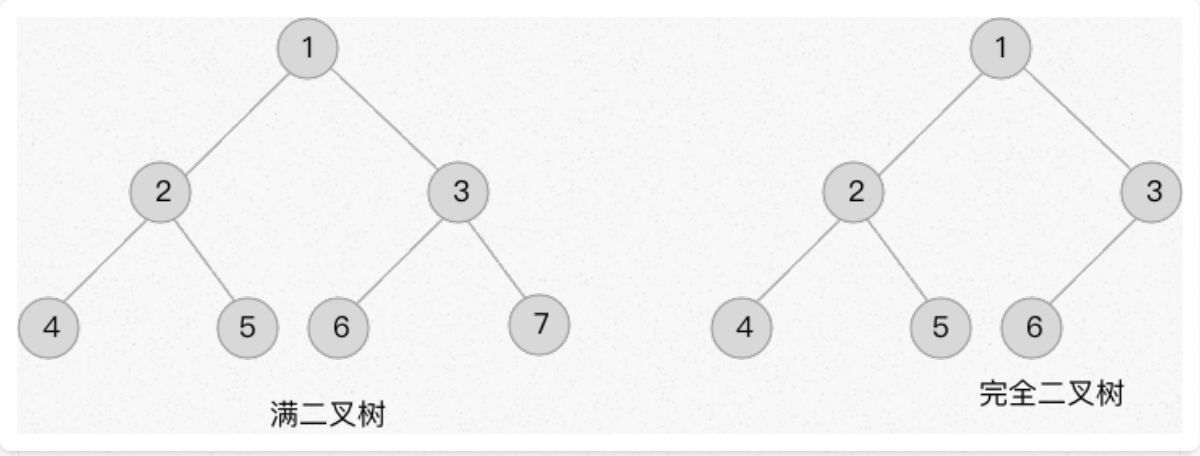
- 堆排序思想（大根堆举例）
  - 按堆的定义将待排序序列r[1..n]调整为大根堆（这个过程成为初建堆），交换r[1]和r[n]，则r[n]为关键字最大的纪录。
  - 将r[1..n-1]重新调整为堆，交换r[1]和r[n-1]，则r[n-1]为关键字次大的纪录。
  - 循环n－1次，直到交换了r[1]和r[2]为止，得到一个非递减的有序序列r[1..n]。
- 所以实现堆排序需要解决两个问题
  - 初建堆，将无序序列构建成堆。
  - 调整堆，去掉堆顶元素，如何将剩余元素调整成一个堆。(初建堆会用到调整堆的操作)。
    - 调整堆：图a是个堆，将堆顶元素97和最后一个元素38交换后，如图b。此时，除根节点外，其余节点均满足堆的性质，由此仅需自上至下进行一条路径上的节点调整即可，调整过程间图c、图d。



    - 建初堆：从最后一个分支节点⌊n/2⌋开始，依次将序号为⌊n/2⌋、⌊n/2⌋-1、...1的节点作为根的子树都调整为堆即可。

树

- 二叉树的性质：
  - 在二叉树的第i层上至多有2^{i-1} 个节点(i≥1)。
  - 深度为k的二叉树至多有2^k -1个节点(k≥1)。
  - 对任何一棵二叉树T，如果其终端节点为n0，度为2的节点为n2，则n0 = n2 + 1。
  - 具有n个节点的完全二叉树的深度为 ⌊ $\log_2 n$ ⌋ + 1。
  - 满二叉树：深度为k且含有2^k - 1个节点的二叉树。
  - 完全二叉树：深度为k，有n个节点的二叉树，当且仅当其每一个结点都与深度为k的满二叉树中编号从1至n的节点一一对应。



- 二叉树的遍历
  - 先序遍历（波兰式）

    1. 访问根节点。
    2. 先序遍历左子树。
    3. 先序遍历右子树。

  - 中序遍历
    1.中序遍历左子树。
    2.访问根节点。
    3.中序遍历右子树。

  - 后续遍历(逆波兰式)
    1.后续遍历左子树。
    2.后续遍历右子树。
    3.访问根节点。

查找

- 折半查找（二分查找），要求线性表必须采用顺序存储结构，而且表中的元素按关键字有序排列。
  - 算法思想
    1. 将给定值key与中间位置纪录的关键字进行比较，若相等则查找成功。
    2. 若不相等则利用中间位置纪录将表对分成前后两个子表。如果key比中间位置纪录的关键字小，则下一次只在前一子表中继续查找，否则在后一子表中继续查找。
    3. 重复步骤1，2，将查找区间不对对分，直到查找成功，或者当前的查找区间为空，则查找失败。
  - 注意：循环执行条件是low ≤ high。
- 二叉排序树（二叉查找树）
  - 定义
    1. 若它的左子树不空，则左子树所有节点的值均小于它的根节点值。
    2. 若它的右子树不空，则右子树所有节点值均大于它的根节点值。
    3. 它的左右子树也称为二叉排序树。
  - 注意：二叉排序树是递归定义的，所以中序遍历一棵二叉树时可以得到一个节点值递增的有序序列。
  - 二叉排序树的递归查找
    - 算法思想：
      1. 若二叉排序树为空，则查找失败，返回空指针。
      2. 若二叉排序树非空，将给定值key与根节点的关键字T->data.key进行比较。

         1.若key等于T->data.key，则查找成功，返回根节点地址。
         2.若key小于T->data.key，则进一步查找左子树。
         3.若key大雨T->data.key，则进一步查找右子树。