1. 排序
   1. 排序算法的分类

1、内部排序：待排序的数据元素全部存入内存中，排序过程中不需要访问外存

//explain：内存的IO性能比外存高的多

1. 插入排序 最好O(n)，平均/最差O(n2)
2. 交换排序
3. 选择排序
4. 归并排序

2、外部排序：待排序的数据元素不能全部装入内存，在排序过程中需要访问外存

//attention：外部排序并不是只在外存中进行

* 1. 插入排序分支详解——希儿排序（最小增量排序）

第一趟将序列分为n/2个长度至多为2的子序列再比较，即子序列中两个相邻元素下标相差n/2，称为增量是n/2

第二趟将序列分为n/4个长度至多为4的子序列，增量为n/4

……直到最后一趟增量为1

Attention：但是最后一趟与普通的直接插入排序虽然相似但是并不相同，因为经过前几趟的排序，整体序列已经比较有序，所以需要交换的次数会大大减少

* 1. 交换排序分支详解

1、冒泡排序

i : 1~n j : 0~n-i

优化：三角的最后上角有可以跳过的地方，即“最后一次没有发生交换时”，就可以退出了，这样最后一趟不需要跑

1. 快速排序

任选一个元素，把所有元素和它比较，把所有比它小的元素排在前面，所有比它大的元素排在后面；之后划分为两个部分，分别重复直到每个部分仅有一个元素

每次划分为O(n)，共需要划分log n次，故整体时间复杂度为O(n log n)

一般而言快速排序是目前最好的排序算法，但是对于已经有序的序列却是快速排序的最坏情况，时间退化为简单插入排序，即O(n2)

1. 简单选择排序（不稳定）

//排序算法是否稳定是指比较值相同的两个元素在排序后是否能保持原有顺序

左半部分为有序序列，右半部分为无序序列，每次在右半部分中选择最小的元素然后与无序序列中的最左边的元素交换

* 1. 堆排序

1. 堆的定义：
   1. 堆是满足下列性质的数列{r}：
      1. 小根堆：ri≤r2i+1 ；ri≤r2i+2（即根节点与其左右子节点）
      2. 大根堆：ri≥r2i+1 ；ri≥r2i+2
   2. 堆可以视作一棵完全二叉树（完全二叉树与顺序表的映射），在每个直系路径中父子节点具有大小关系，但是不处于同一直系路径中的兄弟节点不一定具有大小关系
2. 筛选的方法（在输出堆顶元素后将剩余元素调整为一个新的堆）：
   1. 输出堆顶元素之后，以堆中最后一个元素替代
   2. 将根
3. 建堆的方法（将一个无序序列构建成一个堆）：

从最后一个非终端节点（下标是（n-2）/2）开始进行“筛选”，直到形成堆

1. 时间复杂度是O(nlogn)，空间上只需要一个元素的临时存储空间用于交换
   1. 归并排序

将n个元素视为n个长度为1的子序列，然后每次都两两归并，每次得到n/2个各自有序的子序列，最后得到一个整体有序的序列（2路归并）

在内外存共同运行中，最长可以产生2倍内存（memory）的初始顺串

//归并操作其实十分简单，只需要在两个序列中各自比较找出最小的，然后放到另一个长度为两序列之和的序列中。代价是空间占用很大，需要长度为两序列之和的临时空间。

* 1. 基数排序

1. 多关键字排序
   1. 最高位优先MSD

需要逐层分割成子序列然后分别排序

* 1. 最低位优先LSD

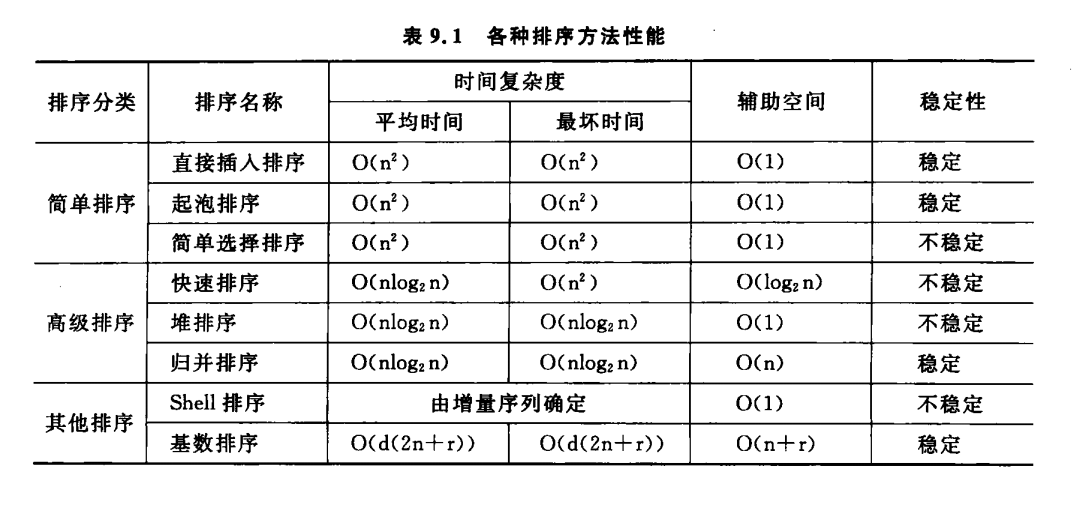
LSD这样的方法不需要分割成子序列，也不需要比较关键字的大小，而是通过“分配”与“收集”来实现

1. 链式基数排序
   1. 原理：使用多关键字思想来进行单关键字排序，比如将各个数字元素拆分为千位、百位、十位、个位，然后多关键字排序
   2. 链式分配与收集

按数位从低到高进行分配，设置0~9十个“桶”，然后将各个元素逐个按元素分配放入桶中再按顺序收集，直到所有数位都分配过之后再收集就是有序的

* 1. 若数组长度为n，基数为r，关键字位数为d，则时间复杂度为O(d(2n+r))，且基数排序是稳定的排序方法
  2. 非链式实现方法：先统计当前数位下
  3. 内部排序算法小结

排序算法性能：



Attention：快速排序和所有选择排序（包括堆排序）都是不稳定的排序