1 内部排序:待排序记录放在内存,排序过程在内存进行
2 外部排序:待排序记录大,排序过程需要访问外存
3 稳定算法:两个关键字相等的记录,排序之后顺序与之前相同,称为稳定排序,否则不稳定
4 时间代价:记录的比较和交换次数, 空间代价:所需附加的空间的大小
5 插入排序: 将待排序的记录插入到有序集合的合适位置
6 第一层循环:依次插入 n 个记录 第二层循环:将第 n 个记录插到合适的位置(逐次交换,不再交换就 **break**)
7 算法是稳定的, 空间代价 $\theta(1)$(需要一个辅助空间用来交换), 最佳情况 $\theta(n)$(n-1 次比较) 最差情况 $\theta(n^2)$(逆序) 平均情况 $\theta(n^2)$
8 记录数量较小时,直接插入排序是高效的排序算法
9 优化的插入排序算法:不用每次都 swap,而只是每次将前面的元素后移,最后再将插入的元素放回应有的位置,减少交换次数,但是算法复杂性没有本质改变,对大规模数据集性能会提高
10 二分法插入排序:二分查找需要插入的位置,算法也是稳定的 比较次数降至 nlogn 量级,移动次数仍为 n*n 量级,最佳情况总时间代价为 nlogn,最差和平均情况下仍为 n*n
11 冒泡排序:不停比较相邻的记录,交换直到所有的记录都排好序
12 稳定的算法,最大最小平均时间代价均为 $\theta(n^2)$
13 优化的冒泡排序:检查每一次冒泡过程中是否发生了交换,没有就结束,最小时间代价优化为 $\theta(n)$
14 直接选择排序:每一次选择后面中最小的元素与 r[i]交换
15 优点:实现简单, 缺点:每一次只能确定一个元素
16 时间效率 $O(n^2)$,空间效率 O(1),不稳定算法
17 简单排序算法的时间代价对比:ppt_p26
18 Shell 排序:基于直接插入排序的两个性质:序列较短时效率高,整体有序时时间代价低
19 先转化成为若干的小序列,在小序列中进行直接插入排序,逐步扩大序列规模,最后进行一次完整的插入排序
20 Hibbard 增量序列:$\{2^k - 1, ..., 7, 3, 1\}$ 排序效率可达 $\theta(n^{3/2})$
21 Shell 排序是不稳定的排序算法
22 快速排序 基于分治思想的快速排序, 轴值选择->序列划分->递归排序 注重分割
23 轴值选择的策略:使 LR 长度尽量相等 固定位置法,中值计算法,随机选择法
24 序列划分的算法:轴值选择(选择最左元素),ij 指针,(左移(j--)交换(i++),右移交换)$_n$,轴值归位,返回轴值位置 排序树
25 算法分析:最佳性能 T(n)=O(nlog(n))
26 最差性能 $O(n^2)$
27 平均性能 T(n)=2/n(sum(T(i)))+cn 得到递推公式 t(n)/n+1 = t(n - 1) / n + 2c / n + 1
28 平均情况下时间复杂度相当好
29 归并排序:简单的划分为两个子序列,对每一个子序列递归划分,最后合并为一个有序序列侧重于合并
30 确保稳定性,归并时左边数组优先 时间复杂度 O(n),一共进行[logn](向上取整)趟,空间复杂度 O(n)是辅助存储量最多的排序方法, 稳定的排序算法
31 堆排序 基于最大(小)堆排序, 建立最大堆(O(n)),取出最大元素放在堆末尾,建堆,循环
32 非稳定排序,总时间代价 $\theta(nlogn)$ 辅助空间代价 $\theta(1)$
33 分配排序:通过分配收集完成排序,前提需要事先知道待排序列的具体情况
34 桶排序:标记为 m 个桶,记录分配到对应桶中,依次收集记录,组成有序序列

35 统计每个取值出现的次数 得到 count 数组, 将其变为前缀和, 在从后向前扫描原数组 array[--count[temparray[i]]] = tmparray[i] (每次减一因为记录的是它的后继的起始下标,取前置自减因为是后继的下标)
36 从后向前扫描保证是稳定的排序算法 时间代价 θ (m+n) 空间代价 θ (m+n),m 个计数器,n 个临时空间(tmparrray) 适用于 m 相对于 n 很小的情况, 是一种稳定的算法
37 桶式排序只适用于 m 很小的情况
38 基数排序:每一个排序码由 d 位子排序码组成,ki 的取值范围称为基数,记为 r
39 高位优先法:先排高位,再排低位 低位优先法:先排低位,再排高位
40 高位优先法:位由高到低分次排序,分分分分分兽的过程, 是一个递归分治问题
41 低位排序法:从最低位到最高位分次排序,是一个分收分收的过程,简单,计算机常用
42 基数排序的实现:基于静态存储,基于静态链存储
43 基于数组的基数排序算法:不断进行桶排序,排序码为 array[i]/radix % r
44 空间代价:临时数组,r 个计数器-> θ (n+r) 时间代价:桶式排序 θ (n+r),进行 d 次 θ (d(r+n))
45 基于静态链的基数排序:分配出来的子序列存储在 r 个静态链组织的队列中,避免移动所有记录
46 静态队列定义:结点:关键码值+后继结点下标 静态队列类:头指针,尾指针
47 算法:建链:初始指向下一个元素,链尾 next 为空 对 i 个排序码分配收集
48 分配算法:初始化 r 个队列,对整个静态链分配,每一个元素加到对应链的尾部,之后分配下一个记录
49 收集过程:找到第一个非空队列,维护整体的 first,last 下标,循环收集下一个非空的队列,首位相连,更新 last 下标,完毕,尾指针 next 域为零
50 空间代价 n 个记录指针,r 个头尾指针,O(n+r), 时间代价仍是 O(d(n+r)),因为 d 大于等于 logn 空间复杂度仍为 O(nlogn)
51 索引排序