**算法导论**

**几种常见的排序算法：**

算法稳定是指相等的数据排序后相对位置不变

几种简单交换相邻元素的排序算法复杂度高是因为一次只减少一个逆序，而另外几种通过比较距离较远的元素可以一次减少更多逆序！

1.简单插入排序

时间复杂度为o（n\*n），空间复杂度为o（1）,稳定

算法思想是先排两个，然后将第三个加入排好的长度为2的序列（插入的过程是从已经排序的序列后面开始比，将大的数往后挪，直到某个数比要插入的数小，在这个位置插入要插入的数），然后将第4、5、6。。。后续的所有的数逐个加入已排序的序列，过程像是不断摸扑克牌并整理。插入排序若输入数据已经预先排序则可以在o（N）时间内运行完成，因此若预排序情况较好则插入排序性能不错，这也是shell排序的原理

2.冒泡排序

时间复杂度为o（n\*n），空间复杂度为o（1），稳定

算法思想是遍历一遍，将最大的数挪到最后面，然后其余n-1个数再遍历再将最小的数挪到最后面，一直循环n-1次

3.简单选择排序

时间复杂度为o（n\*n），空间复杂度为o（1），不稳定

算法思想是遍历一遍找到最小数的位置，将它与第一个数交换，然后遍历一遍找到第二小的数的位置和第二个数交换。。。

不稳定看一个例子：5 8 5 2。第一次选择将第一个5和2交换，导致原本的第一个5变成了第二个5，所以不稳定

4.归并排序

时间复杂度为o（n\*logn），空间复杂度为o（n\*logn），稳定

算法思想有点类似sc译码的过程，排序一个长为n的序列只要将长为n/2的前半部分序列和后半部分序列分别排序，再用一个数组像吸尘器吸两团纸一样将这两个序列从小到大的数吸进来，要得到长为n/2的排序序列就这样递归一直到排长为1的序列。本质是空间换时间，可以并行计算。时间复杂度理解成共需要logn步（比如n个长为1的并为n/2个长为2的过程称为一步）并的过程，每步并需要n的复杂度。稳定性可以通过规定在合并时若两个元素相同则优先选择第一个数组的元素并入来保证

5.快速排序

时间复杂度平均为o（n\*logn），最坏为o（N）,空间复杂度为o（logn）~o(n)，不稳定

算法思想是从值中找一个旗帜，将序列整理为旗帜左边的数都小于旗帜，旗帜右边的数都大于旗帜，然后递归的分别将左右两边的序列排序

整理的步骤就需要设立两个下标lo和hi，分别位于头尾两端，hi向前移动，当hi的值小于旗帜时与lo交换值，然后lo向右移动，当lo的值大于旗帜时与hi交换值，lo和hi的移动直到lo=hi，这一过程看做是两边互相丢沙包，hi将小的沙包丢给lo，确保lo前面的值小于旗帜，lo将大的沙包丢个hi，确保hi后面的值大于旗帜，然后在lo=hi处放入旗帜的值就完成了整理。快排每次划分均衡时复杂度与归并排序一样（只要划分比例为常数比例就算是99：1），不均衡时类似于插入排序

6.希尔排序

时间复杂度为o（n\*logn），空间复杂度为o（1），不稳定

算法思想是插入排序的改进。基本原理是若某个序列较为有序，则插入排序比较简单，希尔排序的关键在于不断整理序列使其较为有序。取一个步长d<n,然后按下标除以d的不同余数分为d个子序列（逻辑划分，并不真正改变小标），对每个子序列进行插入排序（增加序列的有序性）；然后减小d的值重复上述步骤，直到d=1，即进行插入排序。对于简单插入排序来说是稳定的，但是shell排序中的元素可以在各自的子序列中移动，如552879中若一开始步长为2，则第一个5和2交换，使原本的两个5交换了相对位置，所以不是稳定的

7.堆排序

时间复杂度为o（nlogn）,空间复杂度为o（1），不稳定

算法思想是将数组按堆的结构进行理解，借鉴冒泡排序思想，从下往上保证父节点值大于子节点的值，这里注意如果父节点大于两个子节点就不用交换，如果父节点值小于子节点值时要交换元素然后交换的那个子节点为根节点的树要重新进行比较交换，这样能构造一个大小为n的最大顶的堆，然后将堆顶元素和数组末尾的值交换，再构造大小为n-1的最大顶堆不断循环

时间复杂度分析：log2+log3+...+logn与nlogn为同阶函数。一共进行n次的堆顶与末尾元素交换，每次进行最大顶的堆重建时，堆共有k=logn层，进行logn次的比较交换。详情见<https://blog.csdn.net/qq_34228570/article/details/80024306>

8.基数排序

时间复杂度为o（k\*n），其中k为常数，空间复杂度为o（n），稳定

算法思想是从按最低位大小进行排序，将排序后的数组按第二位的大小排序，依次排到按最高位大小排序，所以外层循环次数为最大数的位数。创建10\*n大小的二维list（可以用ArrayList）。每次排序时按数字的第k位数字为i放入第i个list，然后将所有存放的数字依次取出放回原数组。

时间复杂度为遍历n长数组放入桶中再取出，循环最大数的位数次，所以为o（k\*n），空间复杂度为10\*每个桶预留n长，所以是o（n）

9.计数排序

时间复杂度为o（n+k），空间复杂度为o（k），稳定

k为n个整数的取值空间大小，0~k，常用于k相对于n来说较小的时候。

算法的基本思想跟基数排序有点像但是更简单。就是用长为k+1的数组作为k+1个桶，每个桶的索引代表k+1个数字中的一个，然后遍历需要排序的数组，将每个数投入对应的桶中（桶里存这个数字的元素个数，进一步变成桶里存小于等于这个索引的元素个数），最后构造一个与原数组大小相同的数组，从后往前遍历原数组元素放入新数组中，目的是保持稳定性，最后返回新数组

**第一部分 基础知识**

分治法的思想：将原问题分解为几个规模较小但类似于原问题的自问题，递归地求解这些子问题，然后再合并这些子问题的解来建立原问题的解

分治算法中求复杂度时可以用代入法求解递归式。代入法首先要猜解的形式，然后证明这个猜测正确，证明从某个m<n开始，然后代入更大的n证明

猜测我们可以用递归树作图（看出树的层数和每层的代价然后求和）求出大致的界，然后再用代入法证明

主定理：当T(n)=aT(n/b)+f(n)时，如果a，b，f（n）的关系满足某些条件时可以直接得到T（n），比较n的logb（a）次方和f（n）哪个在幂次上更大，就以哪个为主

自由树是连通的无环的无向图

满二叉树中的每个节点是叶子结点或者度为2

完全k叉树是所有叶子节点深度相同，且所有内部节点度为k的k叉树。根节点深度为0，只有三个节点的完全2叉树高度为1

排序的最终结果中各个元素的次序依赖于它们之间的比较的算法称为比较排序，比如归并排序，堆排序，快速排序等，任何比较排序的最差情况下都要o（nlgn）复杂度

基数排序和计数排序还有桶排序不是比较排序，可以达到线性时间复杂度

n！和n的n次方是同阶量