Code刷题总结

注意：刷题要注意回顾知识点，总结算法思考逻辑！！！

# 一、知识点基础要求

数据结构

常见的算法逻辑

编程语言

# 二、针对一个题的常见的思考方向

自我猜测：从数据结构？从常见的算法逻辑？从编程语言的函数？

面对一个习题，我应该思考哪些方面呢？

①首先分析题的输入，输出，确定题属于什么问题？

②根据问题的定性，确定考察的知识性，然后思考如何将知识点应用于解题？

③这个问题可以用什么算法：递归、分治、动态规划、贪心；

# 三、思考全面

在考虑了一般情况后，还要考虑特殊情况，也就是极端值，特殊值等等

# 四、关于某些算法的总结

## 一）递归

递归算法的关键是能够将问题分解为子问题，实质上是分治算法：将原问题分解为多个相互独立的子问题，然后分别求解子问题，最后合并子问题的解得到原问题的解。

因此递归也是要思考原问题可以分解为怎样的子问题？

递归算法的四个特性：

1. 必须有可达到的终止条件，否则程序陷入死循环；
2. 子问题在规模上比原问题小；
3. 子问题可通过再次递归调用求解；
4. 子问题的解应能组合成整个问题的解。

递归算法一定要明确终止条件；

备注：递归算法关键在于分治，要能够将问题分解为循环子问题；把握住临界点，以邻接点来分解问题；例子：剑指offer中的52题字符串匹配，的邻接点就是：下一个字符是否为‘\*’，这决定着下一步如何比较字符；

## 二）回溯法

回溯法（探索与回溯法）是一种选优搜索法，又称为试探法，按选优条件向前搜索，以达到目标。但当探索到某一步时，发现原先选择并不优或达不到目标，就退回一步重新选择，这种走不通就退回再走的技术为回溯法，而满足回溯[条件](https://baike.baidu.com/item/%E6%9D%A1%E4%BB%B6/1783021" \t "_blank)的某个[状态](https://baike.baidu.com/item/%E7%8A%B6%E6%80%81/33204)的点称为“回溯点”。

回溯法注意：

*1、回溯就是对使用过的字符进行标记、处理、处理后的去标记*

回溯法实质是图或者树的深度优先遍历算法。将问题的解空间转化为图或者树的结构表示，然后使用深度优先遍历算法进行遍历，在遍历的过程中记录可行解。

也是利用动态规划或者贪心或者分治的思想来处理问题，然后用回溯法来实现；

例如0-1背包问题：给定一个载重量为W，n个物品，物品的重量为Wi，价值为Vi，1<=i<=n，要求:把物品装入背包，并使包内物品价值最大。

对一件物品，可以分为两种情况，一种是有它的最优解，一种是没它的最优解。

回溯法的实现有两种方式：递归、迭代

子集树、排列数

子集树：当所给问题是从n个元素的集合S中找出满足某种性质的子集时，解空间为子集树；

解为一个空间的子集。

这种情况下就是对每一个维度，分别计算其可能的取值。这是一个递归的逻辑，每一维度就是一次递归

排列树：当所给问题是从n个元素的集合S中找出满足某种性质的排列时，解空间为排列树；

解为空间的一组基。

这就是要求以特定维度开始并固定该维度的情况下求剩余维度的组合，这是一个递归的逻辑。这里对每一个维度也是一次递归。

## 三）排序算法

冒泡：分为两个循环，每层循环都是序列的大小；第一层循环是冒泡的次数，第二层循环是每次冒泡需要比较的次数；

选择：每次从未排序的序列中选择最小的元素，放在剩余序列的‘第一个位置’，也就是已排序列的末尾。

插入：分为两个循环，第一个循环是对要插入的元素；第二个循环，是从插入的元素开始，将它和相邻的前一个元素做比较，直到插入元素到了首个位置或者找到比它小的元素。

希尔排序：是对插入排序的改进，结合分治算法，先将序列分为若干子序列，分别插入排序，然后再进行整体的插入排序。

快速：通过分治法，将序列分为两个子序列，分子序列的时候，就对元素做了比较，将比主元大的放在一侧，比主元小的放在一侧。递归进行，直到子序列个数小于1；

归并：通过分治法，将序列分割为可以直接合并的小子序列，然后合并已经排好学的子序列；这是一个递归算法，先分解，返回有序的子序列，然后合并有序的子序列。

堆排序：首先需要将序列创建为一个堆，然后通过维护堆来排序；维护堆结构，可以通过向上维护，也可以通过向下维护

基数排序：基数排序是一种非比较型整数排序算法，其原理是将整数按位数切割成不同的数字，然后按每个位数分别比较。由于整数也可以表达字符串（比如名字或日期）和特定格式的浮点数，所以基数排序也不是只能使用于整数。

计数排序：

## 四）动态规划

动态规划是求解决策过程的最优化方法。20世纪50年代初[美国](https://baike.baidu.com/item/%E7%BE%8E%E5%9B%BD" \t "_blank)数学家R.E.Bellman等人在研究多阶段决策过程(multistep decision process)的优化问题时，提出了著名的最优化原理(principle of optimality)，把多阶段过程转化为一系列单阶段问题，利用各阶段之间的关系，逐个求解，创立了解决这类过程优化问题的新方法——动态规划。

动态规划一般可分为线性动规，区域动规，树形动规，背包动规四类。

动态规划适用于子问题重复比较多的问题，也就是不同的子问题具有公共的子子问题。

算法基础

五大常用算法之一：分治算法   
<https://www.cnblogs.com/steven_oyj/archive/2010/05/22/1741370.html>

五种常用算法之二：动态规划算法  
<https://www.cnblogs.com/steven_oyj/archive/2010/05/22/1741374.html>

五大常用算法之三贪心算法  
<https://www.cnblogs.com/steven_oyj/archive/2010/05/22/1741375.html>

五大常用算法之四：回溯法  
<http://www.cnblogs.com/steven_oyj/archive/2010/05/22/1741376.html>

五大常用算法之五：分支限界法  
<https://www.cnblogs.com/steven_oyj/archive/2010/05/22/1741378.html>

九大排序算法  
<https://blog.csdn.net/foreverling/article/details/43798223>

二十三种设计模式  
<https://www.cnblogs.com/geek6/p/3951677.html>

# 五、小技巧小记

1、对于整数的操作，有些时候做位相关的操作，可以提高效率

2、双指针法，左右指针可以简化遍历

# 六、python中常用的库

operator：迭代，比较，排序

collections：常用数据结构

random : 随机数

re:正则表达式

math：数学

sys：系统

collections：队列

Queue：队列

heapq：堆

numpy: 矩阵

threading: 进程、线程