数据结构与算法

2018年9月22日 19:48

1. 算法分类
    - 从方法上分
        - **Dynamic Programing 动态规划**
        - **回溯**
        - **递归**
        - **排序（**https://blog.csdn.net/touch_2011/article/details/6767673**）**
        - **二分及其变种（**https://www.cnblogs.com/wsw-seu/p/7681740.html）
        - **LIS（最长上升子序列）**
        - 双指针
            - 快排可以用双指针
            - 链表有些问题会涉及快慢指针
    - 从数据（结构）上分，和该数据结构相关的
        - 字符串
            - 回文
            - **最小编辑距离**
            - 最长公共子串 & 子序列（LCS）
        - **链表**
            - 反转
            - 有序链表合并
            - 复杂链表的复制
            - 链表成环
                - 是否成环
                - 找出环的入口
        - **二叉树**
            - 遍历（前中后序 | 递归&非递归）
            - 判断是否是完全二叉树
            - 求树的高度
            - 已知前序、中序遍历结果（或者其他两种遍历结果），构造二叉树
            - 求二叉树镜像
            - 二叉搜索数中第k大的数
        - 图
            - 图的遍历
            - 拓扑排序
    - 利用数据结构辅助
        - 栈与队列
            - 二叉树打印：z字形打印 | 一行一行打印
            - 带最大值（最小值）的栈结构：O(1)复杂度返回最大值（最小值）

        - BIT（binary index tree）（感觉这个属于比较难的orz）
            - 一般问题都是，给定一个数组，然后求前n个数的和，然后其中涉及改变数组中的变量
2. 经验
    - 加粗部分感觉是比较重要的部分，需要比较熟悉，要了解最基本的解法
    - 一般来说，我自己感觉上
        - DP： 当前状态可以用历史的状态计算得到，因此可以免去一些重复计算。关键是找到迭代的公式：dp[i]和dp[j]之间的关系。有的是一维迭代：dp[i]，经典的就是斐波那契数列（比如贴瓷砖、青蛙跳阶梯）。有的是二维迭代：dp[i][j]，比如一个矩阵左上角到右下角的最短路径（虽然这个可以用一维DP解决），还有字符串的最长公共子串
        - 递归：可以把任务分成多个子任务，比如判断两棵树是否相等，可以分成判断左子树是否相等 + 当前结点是否相等 + 判断右子树是否相等。递归需要注意终止条件。
        - 回溯：我自己感觉回溯更偏遍历的感觉，其中会用到递归调用。一般都会有一个变量保存当前访问过的路径，然后会有一种“从当前节点出发，如果找不到解，再回到当前节点”的意思。经典应用应该是走迷宫？大概就是你所说的“深搜"
    - 这些大概可以有迹可循，多刷些类似的题可以找到一些规律。掌握这些应该就差不多了，基本解法就这些。其他的（我感觉）有点无迹可寻，也有可能是我才疏学浅，总之这些就只能靠多刷题了（吧。
    - **分类型刷题 + 首要选择 《剑指offer》**