结期总结

先给自己点个赞，每天都坚持下来了。

基础的数据结构

刷题的总体顺序设计的非常合理。从**基础的数据结构**入手：数组、链表、字符串、哈希、栈和队列、二叉树……因为我用的是 python，在 python 的运用当中，对于底层的这些数据结构算是省了非常大的力。这些配合上 python 的基础语法，可以说会让每个人的基本功有一定的提升。

这里我想重点说说二叉树。**二叉树**的很多解决方案可以用递归来完成，而递归方案的代码少，但是后面的逻辑却不一定简单。这部分我的理解仍然欠缺，需要多看多练，多多理解。

回溯算法

接下来就是**回溯算法**，也就是大伙常说的暴力法。费力不讨好，但是好用。回溯算法非常好理解，模板也非常清晰：

- 确定函数返回值以及参数，我个人感觉参数在写具体逻辑的时候慢慢添加即可。
- 确定终止条件：这个是非常重要的，不然会让整个判断逻辑无限循环
- 确定遍历逻辑：for(当前层的可能情况): # 解空间的宽度

  Backtracking() # 解空间的深度

在回溯算法中，剪枝非常非常重要，可以极大的优化，减少资源的浪费。

贪心算法

然后就是**贪心算法**了。贪心算法我认为对于数学功底的要求可能要高一些，有一些实际问题往往可以用非常巧妙的数学逻辑解决，非常快捷。比如确立极值点个数就是代表波峰波谷……

但是同样，解决这类题的时候没有固定的模板可以套，并且我们往往需要手动来验证贪心究竟是否成立，在普通条件下自然好说，在极端条件是否仍然成立？

动态规划

然后就是 **DP** 了。动态规划里面涵盖的内容多，应用广。简单的题目一看就懂，仿佛用 DP 都是费力了；但是难题也让人摸不到头脑。

模板如下：

1. 确定 dp 数组（dp table）以及下标的含义
2. 确定递推公式
3. dp 数组如何初始化
4. 确定遍历顺序
5. 举例推导 dp 数组

一头雾水的时候往往需要采取最朴素的方法，也就是 print 大法，但不如手算，这会让对于整个 dp 的逻辑理解提升不少。

单调栈

坦白来说之前我都没有接触过单调栈，不过好在顾名思义，整体理解的难度不大，难在如何与实际题目相结合。

在入栈出栈的时候，我们根据以下三种情况进行判断

- 情况一：当前遍历的元素 T[i]小于栈顶元素 T[st.top()]的情况
- 情况二：当前遍历的元素 T[i]等于栈顶元素 T[st.top()]的情况
- 情况三：当前遍历的元素 T[i]大于栈顶元素 T[st.top()]的情况

这样可以保证栈内元素有序，可以有序的处理实际问题。

图论

图论在某些方面可以结合树一起来看，比如说 BFS,DFS。这些最基础的算法适用于一切有序（不一定是大小，可以是整个数据结构有规律这种样子）的结构。通过内在的逻辑判断，可以找到最短路径，找到所有路径等问题。并且我认为图论中，更会提升对数据结构的理解。因为一切的算法都要以邻接表、邻接矩阵为基础才能进行 coding。

同样，很多题目难度很大，或者说很绕，至今有两题我也没搞懂

检讨

有两天为了打卡而打卡，影响了对于题目的理解和总结的质量。不过当天会进行标注，之后有空重新看过了。

同样，在每天的打卡中，我将自己认为的难点难题记录，方便二刷的时候重新巩固记忆。

感谢 carl，感谢助教，也感谢一起打卡的朋友们。每次看到你们的打卡，让我也产生了动力！也谢谢坚持下来的我自己，还是有一点点成就感！接下来希望找实习顺利。