蓝书Incredible——DA★ZE~

第一章

1.特殊点中位数（P6）

- 如果猜测数值可以试试中位数的特性ovo

2.汉诺塔专项（P27）

- 搬盘子操作可逆...（对称性）
- 所以步数差就很厉害

3.最小值最大 或者反过来 = 二分（P28）

- 因为容易忘记二分，就写到这里啦~
- 写二分对一些问题 等于 **加个限制条件**变成判断问题

4.无根树转有根树 & 深度表（P35）

- 对树的问题还是很有用吧?
- ~~我写的树太少了~~

5.动态维护[维护信息，而不是重新计算]（P42）

- 对某些题目思考一波，可能有些东西可以动态维护减去了一层时间复杂度哦
- 这个题的差值最大也挺有用的/模板qwq

6.Floyd判圈算法（P44）

- 俩人，一个跑的快（一次走两步），另一个跑的慢（一次走一步）
- 如果有环（链表环，循环节.....各种形式的环），那么慢的肯定会被快的追上（速度差1，追及问题）
- lyj大大的代码节选

```
int ans=k;
int k1=k,k2=k;
do{
    k1=next(n,k1);//小孩1
    k2=next(n,k2);if(k2>ans)ans=k2;//小孩2，第一步
    k2=next(n,k2);if(k2>ans)ans=k2;//小孩2，第二步
    //这里的俩if是针对那个题的
}while(k1!=k2);
```

7.1扫描线 & 细节思考（P46）

- 扫描线操作很厉害，转换模型使用很强的!
- 注意所有题目的**恰好**的地方，那些**关键的临界点**!

7.2扫描线 & 悬线法（P51）

- 最大子矩阵...O(nm)复杂度哦
- 单调栈写法qwq

8.枚举所有 -?> 枚举部分（P53）

- 打不了写的暴力，然后考虑优化容易偏到这里23333
- 有些东西不用枚举，可以预处理+靠枚举别的限制O(1)算出来啦

9.降维打击 & 多维前缀和问题（P55）

- **解决高维问题的常见思路是降维**
- P55题目题解1：综合了一些(8)和(7.2)，枚举上下推一维是一些题目的解决技巧
- 多维前缀和，自己瞎写吧233
- P56代码给了个高维容易扩展的版本（but我没看懂）

10.折半查找/Meet-in-the-Middle（P58）

- 本来是2^n的复杂度，能降低到$2^{\frac{n}{2}} * logn$
- 集合拆开对半分，两边分开算数然后枚举一边在另一边查找
- 好多东西都可以对半拆这么玩哈哈哈哈

11.DAG dp（P60）

- 这个感觉能控制点了，方向那种的感觉
- 来到这一条主要靠抽象数学模型哦

12.预定加和/提前算数（P64）

- 就像递推，因为前一个的影响，后一堆都会如何如何。
- 但是后边被影响的那些元素贡献可能分开算不好计算，这时可以提前加他们的被影响贡献到答案，等于他们都不用算那些影响啦！
- 影响合并 vs 影响分开

13.递推约瑟夫环最后留下的人（P65）

- 直接上公式：
  假设编号0~n-1的n个数排成一圈，从0开始每k个数删除一个，最后留下的数字编号记为f(n)
  则f(1)=0,f(n)=(f(n-1)+k)%n
- 看准上边的n是每一个递推的那个n，不是每次都是模最终n
- 原因详见蓝书，简略就是能重新编号
- ex：第一个删除m的话，答案是(m-k+1+f[n])%n，这个数字要改成1~n之间的
  解释一下的话感觉就是相对偏移，还是能重新编号然后算差值

14.LCS（最长公共子序列O(n^2)）转换成LIS（最长上升子序列O(nlogn)）（P66）

- 要求俩序列分别各自序列里都没有相同元素
- 给A序列重新编号1~lenA+1
- 对应的按着A序列对应给B序列编号

- example：
  A={1,7,5,4,8,3,9}
  B={1,4,3,5,6,2,8,9}
  重新编号后的：
  A={1,2,3,4,5,6,7}
  B={1,4,6,3,0,0,5,7}
- 然后在B里求LIS就好啦~（n^2 -> nlogn）

15.在dp里顺手递推计算优化时间（P69）

- 类似（5）

16.枚举一个集合S的子集S0（P70）

- 这太强了！

```
f[0]=0;
int ALL=(1<<n)-1;
for(int S=1;S<(1<<n);S++){
    f[S]=0;
    for(int S0=S;S0;S0=(S0-1)&S)
        if(cover[S0]==ALL)f[S]=max(f[S],f[S^S0]+1);
}
printf("Case %d: %d\n",++kase,f[ALL]);
```

- 简单来讲就是，(S0-1)&S，后边全1正好&之后搞掉一个，加上最后S\^S0正好枚举了所有不一样的0\^1组成的子集
- 太强了太强了，wsl
- 这个时间复杂度有点迷qwq

17.两个优化量的dp & dp状态设计改善方案（P71）

- 两个优化量v1、v2，要求首先满足v1最小，在v1相同的情况下v2最小。
  - 可以把二者组合成一个量M*(v1)+(v2)，其中M是一个比“v2的理论最大值和理论最小值之差”还要大的数。
- dp状态设计影响抉择的话，就都放到状态里试试叭！
- 当前感觉：dp状态后效？ >> 状态修正/决策考虑

18.dp方程——合并与增减变换（P73）

- 给人感觉就像是求 累加$\sum$ 或者是求 累乘$\prod$式子变换一样，无关的搞出来，有关的可以组合——可能新结合量有单调性之类的考虑哦

19.集合dp的状态再压缩（P75）

- 可能有些状态是重复的表示！想想有什么能统一的方法ovo

第三章

##