Fudan ACM-ICPC Summer Training Camp 2015

Team 6 汤定一/马天翼/金杰

2015 年 8 月 18 日

1 概况

本场训练，我们队伍在比赛中完成了 3 道题目，比赛后完成了 3 道题目，共完成 6 道题目。

2 训练过程

开局 mty 写 03，wa 了以后发现没那么简单。jj 写 05，没过样例，喊 tdy 马上上来重写。jj 去读题，发现 04 是水题，准备写。tdy 因为公式推错 wa2 发，ij 打反 wa1 发，74min 通过。jj 觉得好像没写错，把 04 代码交给 mty 看，mty 也觉得题意和代码都没错，于是放弃。tdy 写 01，jj 和 mty 搞 07。tdy 发现公式错了，mty 去写 07。tdy 回来写 01，仍然未过，发现是题意读错，于是再换 07。期间 jj 在读题和想 03 和 08。jj 莫名想到 04 中的一个特例，于是修改后通过，此时还有 2 小时。mty 和 jj 一起看 07 代码找错，终于改对之后提交，TLE。然后 jj 想到 07 一个简单的结论，于是省了一个搜索，但 tdy 在机子上写 01。01 在第四个小时通过，还剩 1 小时。mty 修改 07 之后，wa 了，修改了几次仍然 wa，因为代码是之前的改的，所以很长而且逻辑混乱，jj 决定重写 07，此时还有 40min。tdy 想出了 03，在强调肯定没有错而且写的很快之后，jj 将机子让给 tdy。03 没有过样例，于是 jj 继续重写 07，此时还有 20min。07 没过样例，换 03，此时还有 8min。07 读代码找到错误，在还有 4min 提交了一次，wa 了。剩下 4min 继续尝试 03，至结束未通过。07 在赛后调试 10min 后通过。

3 解题报告

Problem A. Expression

负责 汤定一

情况 比赛中通过 - 238min(3Y)

题意 给定 n 个数的表达式，每次任选一个运算符，把它两边的数按该运算符运算后放回该位置，直到最后只剩一个数。问所有选取的序列得到的答案和。回车不换行。

题解 n^3 动规。从小到大枚举区间的长度，区间的答案等于以每个运算符为最后一个运算符得到的答案和，注意要乘以组合数。

Problem B. Hack it!

负责 汤定一

情况 赛后通过

题意 给定左括号的权值 p，右括号的权值 q，问能否有两个长度相等且小于等于 10000 的仅包含左右括号的串，它们在 base 进制下对 r 取余的值相等。

题解 构造，用两种串 ()()，(()) 构造，即对 2500 个位置进行构造，()() 为 w1，(()) 为 w2，若 w1>=w2，则记为 ci=w1-w2，否则为 ci=w2-w1，我们发现 2500 个位置有 2500 个 ci 值，能乘 -1,0,1，每次取最大的两个值，把它们的差插回去，看有没有相等的值，一直做到有相等的为止，再构造答案即可。

Problem C. GCD Tree

情况 尚未通过

Problem D. Too Simple

负责 金杰

情况 比赛中通过 - 175min(4Y)

题意 有 m 个从集合 1-n 到集合 1-n 的映射，有一些映射不知道，表示成 -1，问最后有多少方案能使所有的 i 经过所有映射后等于 i。

题解 如果有多个 -1，则其他 -1 随便排，让最后一个 -1 排成满足答案的即可。如果没有 -1，则直接判断是否满足答案。注意，即使有 -1，如果某个映射使集合 size 变小了也是永远不能满足答案的。

Problem E. Arithmetic Sequence

负责 汤定一

情况 比赛中通过 - 74min(4Y)

题意 给定 n 个数的数列。给定 d1、d2，若一个区间满足区间由以 d1、d2 两个等差数列连接而成，则答案 +1，问答案为多少。

题解 当 d1!=d2 时，以每个点转折点统计答案，当 d1=d2 时，找到满足条件的区间长度 len，答案加上 len*(len+1)/2 即可。

Problem F. Persistent Link/cut Tree

负责 汤定一

情况 赛后通过

题意 给定 m 棵树，每棵树 i 都是由 ai(ai<i) 的 x 结点和 bi(bi<i) 的 y 结点以权值 li 的边连接起来。问每棵树的所有点对的距离和。

题解 两个树合起来的答案等于它们分别的答案加上以 x 为根和以 y 为根的树以 li 为边权合起来的值。有两种询问，1. 以点 x 为根的树所有点到 x 点的距离和，因为每颗数都是由两棵树合成的，可以用类线段树的思维计算出。2. 一棵树中两个点的距离，若 x、y 在同一子树中，则继续递归下去，否则为 x 到 ai 子树的根的距离加上 y 到 bi 子树的根的距离加上 ai 连接 bi 的边权，用记忆化搜索即可。

Problem G. Travelling Salesman Problem

负责 金杰、马天翼

情况 比赛后通过

题意 n*m 的棋盘上每个点都有一个非负的数，问从左上角走到右下角最大能收集到的数字和。

题解 如果 n 或 m 是奇数，则走蛇形即可。否则至少有 1 个点会到不了。如果 i+j 为奇数，发现一定能构造只牺牲这个点的方案。如果 i+j 为偶数，发现至少牺牲另一个 i+j 为奇数的点，所以绝对会经过所有偶数点。于是找奇数点里值最小的即可。

Problem H. Goldbach's Conjecture

情况 尚未通过

Problem I. Random Inversion Machine

情况 尚未通过

Problem J. Sometimes Naive

情况 尚未通过

4 总结

01 虽然也是一个人啃了很多时间，但是每隔一段时间都有一个重大发现，所以没有发生在一个阶段卡半小时的现象，是没有问题的。

04 看起来花掉的时间不多，其实只有 5min 的代码，是特殊情况没有考虑到，在刚开局还有很多陌生题的情况下两个人去找有没有问题而花去很多时间才是失误。

07 的问题很多，首先是在一个人没有出解的情况下就将思路告诉另一个人一起想，所以两个人直接在思考「这种格子怎么走才最省」这样的子问题，其实完全可以避免进入这个问题，所以在没有头绪之前，最好独立思考。

然后是代码问题，07 因为一改再改，中间还套个搜索，然后又加输出方案什么的，代码相当乱。而且代码重用的很少，复制了好多次，每次修改都要这里改那里改，100 行的代码写成了 300 行。所以宁愿多花一点时间，把代码写的漂亮一点，如果错了找错改错反而更节省时间，出现低级错误的概率也会变小。说到底写代码的时间是真的不长的，能减少找错的时间才是王道。

听队友思路时不能嗯嗯嗯的就过去了，没听懂就说出来，否则不如直接写。