LUT 2023ICPC 校赛题解

沈宇昊

兰州理工大学

May 8, 2023

Content

Contents

如果你已经写过 ICPC 区域赛的题，去看过解题报告，或者学某些知识看别人的博客时，往往会碰到别人三言两语讲完一个思路，自己还是一头雾水的情况。

但竞赛的状态就是如此，大部分时候并没有十分精巧且详略得当的最优攻略。

但无需因此止步，或寻求事无巨细的教程而踟蹰不前，或浮夸潦草，人云亦云，复制题解，当个报菜名的大师。二者都是不可取的。

这份题解也是同样的道理，只会告诉你解题的方法，但是不会将方法教给你。等待教学固然是一个途径，但是自己有了需求再去努力会更得心应手

Contents

编译期 a[i] 的形式会被翻译成 *(a+i)，展开后不难发现是三次映射

得到 i[a][a][a] 等价 a[a[a[i]]]

Contents

如果你不知道图论的基本概念

如果你不会存图

原题地址是 atcoder292d，可以去做一下

题面出了重大问题，是要考虑重边和自环，而不是不要考虑，原来的明显和样例冲突了

正常 dfs 一遍，把各连通块的点数和边数数一下即可

代码，仅作参考，对于新手而言还是值得自己写一写的

Contents

动态维护两个有序的数列，很容易想到 set 和二分。
剩下的，看代码吧....

代码

Contents

树状数组、线段树，甚至因为单点修改的缘故 ST 表。。一堆都能写。

但是有这个必要吗?

考虑一前一后两个数，对于任何大小的 L，如果后面的数字更大，那么无论什么情况前面的数都不可能成为答案。

自然的，考虑维护一个答案候选序列，从前往后值依次减小，下标依次增大 (单调队列)。

查询的时候因为单调性，二分一个位置即可，不会单调队列的可以看群里之前的课件和录屏

单调队列 + 二分答案,$O(M + M\log M)$，代码

是否有更简单的做法?

单调队列 + 二分答案,$O(M + M\log M)$，代码

是否有更简单的做法?

考虑并查集，维护单调队列的时候，被更新的值的父亲指向更新他们的值。以此类推，就起到了一个传递的作用。复杂度 O(M)

自己动手试试!

Contents

01trie 模板题，请自行在 OI-wiki 学习 trie 树

trie 的节点表示状态，而边则是转移的方式。考虑数的二进制，从高到低考虑位数，我们很自然的维护了一个已有的数字的 trie，存在各个节点中。

根据异或同 0 异 1 的原则，我们拿到新的数字 x，从高到低的在 trie 中找每一位和 x 不同的数字，如果找不到则找相同的情况，这必然存在，因为二进制上不是 0 就是 1，考虑 32 位，肯定会被保存下来。

同时因为高位最优，自然满足一个贪心的原则

代码

Contents

问朋友要的题，一开始把式子看颠倒了。事实上和整除分块没有什么关系，如果交换分子分母好像是个数论分块 + 前缀和...

枚举 $\sum$ 里的东西，然后确定 k 的上下界算个贡献即可.

```
long long get(int n) {
long long res = 0;
  for (int m = 1; 114 * m * m <= n; m++) {
    int l = 114 * m * m,
        r = min(114 * (m + 1) * (m + 1) - 1, n);
    res += (r - l + 1) * m;
  }
  return res;
}
```

```
long long get(int n) {
long long res = 0;
  for (int m = 1; 114 * m * m <= n; m++) {
    int l = 114 * m * m,
        r = min(114 * (m + 1) * (m + 1) - 1, n);
    res += (r - l + 1) * m;
  }
  return res;
}
```

求一个 $\mathtt{get}(n+514)-\mathtt{get}(n)$ 就好了

Contents

原题 CSP-S 2019 Day1 T2 括号树

自己找题解看吧.. 写不了就放了

关键在于对一道题的尝试，以此题为例，如果一开始没有任何的思路，那么先考虑纯链的情况，然后是尝试自己写几个样例，试着找找树上的规律。尽可能的去找方向思考，而不是一时半会儿想不到正解就直接去看答案，这样对训练也没有什么效果可言

Contents

总结

如果你不会 STL，对各种数据结构不熟悉，也不了解 C++ 的各种写法，递归写的也不熟练，甚至不会，这都不是什么大问题。或者看书，或者按洛谷、acwing 等 OJ 的题单教程来熟悉，都是可以的

基础的内容没什么好" 学" 的，主要靠写题来熟练。

真正重要的是克服恐惧心理，勇敢尝试去理解一些陌生的，抽象的概念，发掘其中的乐趣，而不是找借口，或者回避，纵使不打竞赛，干别的事情，搞项目，学技术，也是同样的道理。

至于平时的训练，该不该看题解。一道题要想多久看题解，诸如此类的问题，自己写的时间久了自然会有答案，我个人的建议是思考 15 分钟左右。当然还是酌情，掌握了一定知识、套路的情况下可以多想一想。

另外，训练的时候要避免眼高手低。切忌没有怎么训练就搞总结性发言，动辄不做模板题，想着要练所谓的“思维”，然而真要从空的 cpp 开始 ac 一个套路题，又不能成功，这不是一件好事。

希望这场比赛对你有所帮助