2024 江苏省大学生程序设计大赛 题解

2024 Jiangsu Collegiate Programming Contest Tutorial

电子科技大学

UESTC

2024 年 5 月 12 日

JSCPC Jiangsu Collegiate Programming Contest

Problem A - 两人成伴, 三人不欢

题目大意

给你一个森林, 你要添加一些边使得这个图双连通并且没有三元环, 问最少添加几条边, 不能输出 -1 $(1 \leq N \leq 10^5)$, 需要输出添加的边的方案.

Problem A - 两人成伴, 三人不欢

- 首先我们考虑一棵树的情况, 如果叶子数量为 m, 每个叶子节点至少要连一条边, 添加的边尽量是叶子之间的, 至少需要 $\lceil m/2 \rceil$ 条边, 同时每条树边需要至少需要与一条新增的边形成环才能保证它最后不成为桥, 我们可以对树进行 dfs 标号, i 号叶子和 $i+m/2$ 连边, 这样就保证了每条树边一定被覆盖到了.

Problem A - 两人成伴, 三人不欢

- 首先我们考虑一棵树的情况, 如果叶子数量为 m, 每个叶子节点至少要连一条边, 添加的边尽量是叶子之间的, 至少需要 $\lceil m/2 \rceil$ 条边, 同时每条树边需要至少需要与一条新增的边形成环才能保证它最后不成为桥, 我们可以对树进行 dfs 标号, i 号叶子和 $i+m/2$ 连边, 这样就保证了每条树边一定被覆盖到了.
- 这样连边会在一种情况下有问题: 如果有 x 个叶子节点的父亲相同, 且 $2x > m$, 那么这些叶子间是不能连边的, 这种 case 下我们让其他的 $m-x$ 个叶子 i 还是和 $i+m/2$ 连边, 剩余未连边的 $2x-m$ 个叶子找因为不能互相连边, 每个找外部的某个叶子连边, 总连边数为 x.

Problem A - 两人成伴, 三人不欢

- 然后我们考虑多棵树的情况, 如果有 k 棵树, 无论最后连边情况怎么样, 我们都可以找出 $k-1$ 条新增的边 $+$ 原来的森林变成一棵树, 然后剩下新增的边其实是在解决一个树上的问题.

Problem A - 两人成伴, 三人不欢

- 然后我们考虑多棵树的情况, 如果有 k 棵树, 无论最后连边情况怎么样, 我们都可以找出 $k-1$ 条新增的边 + 原来的森林变成一棵树, 然后剩下新增的边其实是在解决一个树上的问题.
- 由前面树的 case 可以知道, 为了边数尽量少, 我们希望这棵树上对应的 x 尽量小. 我们考虑如何添加这 $k-1$ 条边, 一个显然的贪心方式是找 x 最大的树, 连接它的的叶子和其他某棵树, 使 x 减 1, 然后再依次连接新的树进来, 于是我们可以按 x 从大到小对 k 棵树排序, 然后维护 x 的优先队列, 每次新增一棵树后, 把所有叶子节点加进优先队列. 最后合并完 k 棵树后, 在按单棵树的做法做一遍即可.

Problem A - 两人成伴，三人不欢

- 对于 2 个点/菊花图等特殊 case，一个比较好的处理方式是，跑完上面的合并流程后，直接使用找三元环的 $|E|\sqrt{|E|}$ 算法判断掉就好
  ps：本题原来给的并不是多棵树，而是一个任意的无向图，由于会使得代码量急剧增加，改成了现在的简化版

Problem B - 恶魔的面积

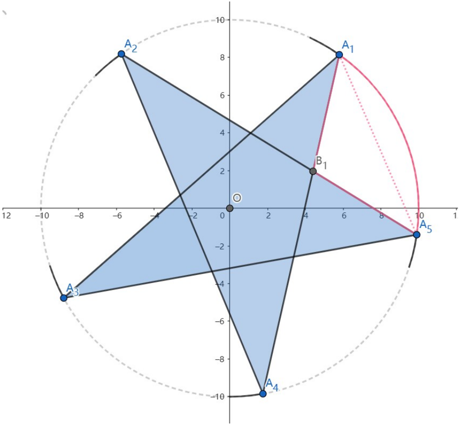
题目大意

给一个圆上的五段不相交的圆弧, 从每个圆弧依次选择一个点然后按顺序连成一个五角星, 求五角星的面积并.

Problem B - 恶魔的面积

- 题目可以转化为求半径为 r 的圆里非恶魔封印点的点集面积, 再用圆面积减去这部分面积. 观察可以发现, 题目转化后要求部分的面积可以分为五个部分, 每个部分是由一个**弓形**加一个三**角形**构成.

Problem B - 恶魔的面积



Problem B - 恶魔的面积

例如要求出 A_1 和 A_5 之间的非恶魔封印点面积, 设线段 $A_{1s}A_{4t}$ 和线段 $A_{5t}A_{2s}$ 的交点为 B_1 , 其中 $A_i s$ 表示第 i 段圆弧的极角最小点, $A_i t$ 表示第 i 段圆弧的极角最大点, 则只需要求出由圆弧 A_5A_1 , 线段 $A_{1s}B_1$ 和线段 B_1A_{5t} 围成的区域即可. 三角形部分可以利用线段求交点的方法求出 B_1 , 再用叉积求出面积, 而弓形则是扇形减去三角形, 简单数学计算即可. 其余部分求法同理. 一个测试数据复杂度 $O(1)$, 总复杂度 $O(T)$.

Problem C - 无线电测向运动

题目大意

交互题. 给一个 n 个点的环（n 为奇数），环上有两个隐藏点. 你需要用不超过 40 次询问来找到这两个隐藏点, 每次询问返回两个隐藏点到询问点的最短距离之和.

Problem C - 无线电测向运动

- 设 $f(x)$ 表示询问 x 点对应的 $dist$ 值.

Problem C - 无线电测向运动

- 设 $f(x)$ 表示询问 x 点对应的 $dist$ 值.
- 考虑 $f(x)$ 在环上的分布情况: 在两点之间时的值都相同, 其中一侧一直在增加, 另一侧在减少.

Problem C - 无线电测向运动

- 设 $f(x)$ 表示询问 x 点对应的 $dist$ 值.
- 考虑 $f(x)$ 在环上的分布情况: 在两点之间时的值都相同, 其中一侧一直在增加, 另一侧在减少.
- 将环拆成两半, 观察 $f(x)$ 在 $1 \sim \lceil \frac{n}{2} \rceil$ 上的变化;

Problem C - 无线电测向运动

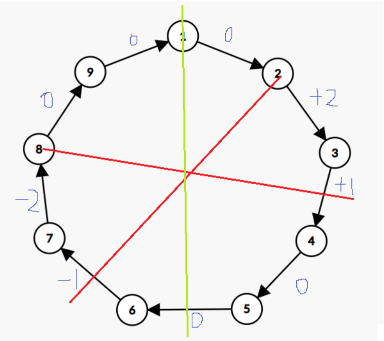
- 设 $f(x)$ 表示询问 x 点对应的 $dist$ 值.
- 考虑 $f(x)$ 在环上的分布情况: 在两点之间时的值都相同, 其中一侧一直在增加, 另一侧在减少.
- 将环拆成两半, 观察 $f(x)$ 在 $1 \sim \lceil \frac{n}{2} \rceil$ 上的变化;
- 可以发现, 一定存在一个点 x, $f(1)$ 到 $f(x)$ 的变化与 $f(x)$ 到 $f(x+1)$ 的变化不同.

Problem C - 无线电测向运动

- 设 $f(x)$ 表示询问 x 点对应的 $dist$ 值.
- 考虑 $f(x)$ 在环上的分布情况: 在两点之间时的值都相同, 其中一侧一直在增加, 另一侧在减少.
- 将环拆成两半, 观察 $f(x)$ 在 $1 \sim \lceil \frac{n}{2} \rceil$ 上的变化;
- 可以发现, 一定存在一个点 x, $f(1)$ 到 $f(x)$ 的变化与 $f(x)$ 到 $f(x+1)$ 的变化不同.
- 故可以二分找到 x

Problem C - 无线电测向运动

- 例如下图：隐藏点为 2 和 8，边上的蓝色数字表示 $f(x+1)-f(x)$ 的值.



Problem C - 无线电测向运动

- 找到 x 之后, x 可能不是隐藏点, 而其对应的隐藏点可能需要绕半圈, 在这个地方上需要稍微讨论一下.

Problem C - 无线电测向运动

- 找到 x 之后, x 可能不是隐藏点, 而其对应的隐藏点可能需要绕半圈, 在这个地方上需要稍微讨论一下.
- 找到其中一个隐藏点之后（设为 N_1），另一个隐藏点（设为 N_2）可以先询问 $f(N_1)$, 这样可以找到 N_2 与 N_1 的距离, 然后再看 N_2 在 N_1 的顺时针还是逆时针, 可以询问 N_1 两边的点得出.

Problem C - 无线电测向运动

- 找到 x 之后, x 可能不是隐藏点, 而其对应的隐藏点可能需要绕半圈, 在这个地方上需要稍微讨论一下.
- 找到其中一个隐藏点之后（设为 N_1）, 另一个隐藏点（设为 N_2）可以先询问 $f(N_1)$, 这样可以找到 N_2 与 N_1 的距离, 然后再看 N_2 在 N_1 的顺时针还是逆时针, 可以询问 N_1 两边的点得出.
- 不难证明, 次数上界为 $\log_2(n/2) + C \approx 35$, 不会超过 40 次.

Problem D - 都市摩天楼

题目大意

给定一个 n 行 n 列的网格，从上到下将行编号为 1 到 n，从左到右将列编号为 1 到 n，初始每个单元格中的值均为 -1. 你需要给单元格赋值，每轮你可以选择一个单元格，考虑与其四连通（上下左右）的单元格（不包括你选择的单元格）中的值组成的集合 S，则你可以将这个单元格重新赋值为 $[0, \text{mex}(S)]$ 中的一个整数，一个格子可以多次进行赋值操作. 请构造一种赋值方案，使得最后值为 3 的单元格数不少于 $\lfloor \frac{2(n-1)^2}{3} \rfloor$ 个.

Problem D - 都市摩天楼

- 考虑一下最后能变成 3 的格子的本质, 那么一个格子能变成 3 的条件是, 相邻四个格子里, 有 ≥ 3 个格子要么最后不是 3, 要么变成 3 的时间在它之后.

Problem D - 都市摩天楼

- 考虑一下最后能变成 3 的格子的本质, 那么一个格子能变成 3 的条件是, 相邻四个格子里, 有 ≥ 3 个格子要么最后不是 3, 要么变成 3 的时间在它之后.
- 这样启发我们去构造一个森林, 每棵树与边界相邻的点作为根节点, 这样我们每棵树按 dfs 序去变成 3 就可以.

Problem D - 都市摩天楼

- 考虑一下最后能变成 3 的格子的本质, 那么一个格子能变成 3 的条件是, 相邻四个格子里, 有 ≥ 3 个格子要么最后不是 3, 要么变成 3 的时间在它之后.
- 这样启发我们去构造一个森林, 每棵树与边界相邻的点作为根节点, 这样我们每棵树按 dfs 序去变成 3 就可以.
- 那么一种比较自然的方法就是, 对于 $i \equiv 0 \pmod 3$ 的行, 从 $(i, 1)$ 一直连到 $(i, n-1)$ 然后 $i-2$, $i-1$ 两行黑白染色, 选一半的格子变为 3, 这样总的比例是 $\frac{1}{3} + \frac{2}{3} \times \frac{1}{2} = \frac{2}{3}$ 左右, 可以满足 $\frac{2(n-1)^2}{3}$ 的限制.

Problem E - 序列操作

题面描述

- 长度为 n 的序列 $a_1, a_2, ..., a_n$.
- q 次询问 l, r, k, 将 $[l, r]$ 区间执行 k 次 Reduce 操作后的最大值, 每次 Reduce 操作将区间最大值整除 2.
- $n, q, a_i \le 10^5$

Problem E - 序列操作

- 考虑对于每次询问，首先二分答案 ans ，变成查询需要多少次操作使得区间最大值 $\le ans$.

Problem E - 序列操作

- 考虑对于每次询问，首先二分答案 ans ，变成查询需要多少次操作使得区间最大值 $\leq ans$.
- 对于每个 a_i, 将不断进行操作的所有取值都看做二维平面的点 $(i, a_i), (i, \lfloor \frac{a_i}{2} \rfloor), (i, \lfloor \frac{a_i}{4} \rfloor), ..., (i, 0)$ 总共 $O(n \log_2 V)$ 个点，那么每次查询就是查询满足 $l \leq x \leq r, ans < y$ 的点 (x, y) 的个数，即操作次数.

Problem E - 序列操作

- 考虑对于每次询问，首先二分答案 ans，变成查询需要多少次操作使得区间最大值 $\le ans$.
- 对于每个 a_i，将不断进行操作的所有取值都看做二维平面的点 $(i, a_i), (i, \lfloor \frac{a_i}{2} \rfloor), (i, \lfloor \frac{a_i}{4} \rfloor), \dots, (i, 0)$ 总共 $O(n \log_2 V)$ 个点，那么每次查询就是查询满足 $l \le x \le r, ans < y$ 的点 (x, y) 的个数，即操作次数.
- 通过主席树或者整体二分等方法维护二维数点，复杂度 $O(n \log^2 V)$. 该算法已足够通过该题.

Problem E - 序列操作

- 考虑将所有 $\frac{a_i}{2^j}$ 按其二进制最高位分层，每层计算前缀个数和后，可枚举判定每个询问位于第几层，将原问题拆分为 $\log V$ 个子问题.

Problem E - 序列操作

- 考虑将所有 $\frac{a_i}{2^j}$ 按其二进制最高位分层, 每层计算前缀个数和后, 可枚举判定每个询问位于第几层, 将原问题拆分为 $\log V$ 个子问题.
- 同时观察: 一个有序数列 s 全部除以 2 后仍保持有序, 我们可以为每个 $\frac{a_i}{2^j}$ 指定同一个序 r_i, 使得每个子问题都可以用该 r_i 比较大小, r_i 可通过逐层插入平衡树等方式在 $O(n \log n)$ 的时间内求出.

Problem E - 序列操作

- 考虑将所有 $\frac{a_i}{2^j}$ 按其二进制最高位分层，每层计算前缀个数和后，可枚举判定每个询问位于第几层，将原问题拆分为 $\log V$ 个子问题.
- 同时观察：一个有序数列 s 全部除以 2 后仍保持有序，我们可以为每个 $\frac{a_i}{2^j}$ 指定同一个序 r_i，使得每个子问题都可以用该 r_i 比较大小，r_i 可通过逐层插入平衡树等方式在 $O(n \log n)$ 的时间内求出.
- 利用 r_i 对所有子问题一起整体二分，可将二分规模缩减到 n，每次检查是大小为 $O(n \log V)$ 的二维数点，使用树状数组统计可得到 $O((n+q) \log n \log \log V)$ 的复杂度. 空间复杂度稍作优化可降至 $O(n+q)$.

Problem E - 序列操作

- 考虑将所有 $\frac{a_i}{2^j}$ 按其二进制最高位分层，每层计算前缀个数和后，可枚举判定每个询问位于第几层，将原问题拆分为 $\log V$ 个子问题.
- 同时观察：一个有序数列 s 全部除以 2 后仍保持有序，我们可以为每个 $\frac{a_i}{2^j}$ 指定同一个序 r_i，使得每个子问题都可以用该 r_i 比较大小，r_i 可通过逐层插入平衡树等方式在 $O(n\log n)$ 的时间内求出.
- 利用 r_i 对所有子问题一起整体二分，可将二分规模缩减到 n，每次检查是大小为 $O(n\log V)$ 的二维数点，使用树状数组统计可得到 $O((n+q)\log n\log\log V)$ 的复杂度. 空间复杂度稍作优化可降至 $O(n+q)$.
- 在 popcount 等位操作复杂度视为常数的前提下，使用 bitset 可将理论复杂度优化至 $O((n+q)\log n)$.

Problem F - 下载速率显示器

题目大意

给出 n, k, 你需要对每个长度为 k 的区间, 计算其平均速率. 如果大于或等于 1024, 则以 MiBps 显示. 否则, 以 KiBps 显示.

- 用双指针遍历每一个长度为 k 的区间然后进行计算模拟即可. 时间复杂度 $O(n)$.

Problem G - 下载时间显示器

题面描述

网络带宽为 B, 两条流先后加入, 第一条流 t_1 时加入, 流大小 a_1, 第二条流 t_2 时加入, 流大小 a_2, 求两条流的 FCT（流完成时间）的理论值. 保证第二条流在不早于第一条流开始时加入.

tag: 讨论

Problem G - 下载时间显示器

- 求 FCT 时，墙钟时间（绝对时间）可以转化为相对时间考虑．也就是令 $t_1 = 0, t_2 \leftarrow t_2 - t_1$．下文中均按相对时间来考虑计算．由于量纲统一，因此下文也不会再强调单位．
- 分如下两种情况考虑

Problem G - 下载时间显示器 - 两条流传输过程没有重叠

- 当第一条流传输结束后，第二条流才开始传输，也就是 $t_2 \geq \frac{a_1}{B}$ 时，可以判定两条流传输过程没有重叠.
- 此时两条流均可以充分利用网络带宽，所以两条流的 FCT 分别为 $\frac{a_1}{B}, \frac{a_2}{B}$.

Problem G - 下载时间显示器 - 两条流传输过程有重叠

- 当第二条流还在第一条流传输时开始传输, 也就是 $t_2 < \frac{a_1}{B}$ 时, 可以判定两条流传输过程有重叠. 此时两条流在某些时刻平分网络带宽, 仍需分两种情况考虑.

Problem G - 下载时间显示器 - 两条流传输过程有重叠

第二条流结束时第一条流未结束

- 为了判定这一点, 我们先求第二条流的 FCT. 此时第二条流全程带宽均为 $\frac{B}{2}$, 因此第二条流的 FCT 为 $\frac{2a_2}{B}$.

Problem G - 下载时间显示器 - 两条流传输过程有重叠

第二条流结束时第一条流未结束

- 为了判定这一点, 我们先求第二条流的 FCT. 此时第二条流全程带宽均为 $\frac{B}{2}$, 因此第二条流的 FCT 为 $\frac{2a_2}{B}$.
- 当第二条流开始时, 第一条流已经传输了 Bt_2 大小的数据, 在第二条流传输的过程中, 第一条流传输的数据量为 $\frac{2a_2}{B} \times \frac{B}{2} = a_2$. 当第二条流结束时, 第一条流理论可以传输 $Bt_2 + a_2$ 大小的数据. 由于此时第一条流仍未结束, 因此判定条件为 $a_1 \geq Bt_2 + a_2$.

Problem G - 下载时间显示器 - 两条流传输过程有重叠

第二条流结束时第一条流未结束

- 为了判定这一点, 我们先求第二条流的 FCT. 此时第二条流全程带宽均为 $\frac{B}{2}$, 因此第二条流的 FCT 为 $\frac{2a_2}{B}$.
- 当第二条流开始时, 第一条流已经传输了 Bt_2 大小的数据, 在第二条流传输的过程中, 第一条流传输的数据量为 $\frac{2a_2}{B} \times \frac{B}{2} = a_2$. 当第二条流结束时, 第一条流理论可以传输 $Bt_2 + a_2$ 大小的数据. 由于此时第一条流仍未结束, 因此判定条件为 $a_1 \geq Bt_2 + a_2$.
- 对于第一条流的 FCT, 由于每时每刻带宽均全部利用, 可以看成两条流数据全部传输完成的时间即为第一条流的 FCT, 因此第一条流的 FCT 为 $\frac{a_1+a_2}{B}$. 同样也可以分作三段（第二条流开始前, 和第二条流一起传输, 第二条流完成后）分别计算, 结果是一样的.

Problem G - 下载时间显示器 - 两条流传输过程有重叠

第二条流结束时第一条流已结束

- 此时也就是 $a_1 < Bt_2 + a_2$ 的情况. 对于第一条流需分成两段: 第二条流加入前和第二条流加入后. 第二条流加入时, 第一条流剩余 $a_1 - Bt_2$ 大小的数据需要传输, 这些数据都会以 $\frac{B}{2}$ 的速率传输, 因此第一条流的 FCT 为 $t_2 + \frac{a_1 - Bt_2}{\frac{B}{2}} = \frac{2a_1 - Bt_2}{B}$.

Problem G - 下载时间显示器 - 两条流传输过程有重叠

第二条流结束时第一条流已结束

- 此时也就是 $a_1 < Bt_2 + a_2$ 的情况. 对于第一条流需分成两段: 第二条流加入前和第二条流加入后. 第二条流加入时, 第一条流剩余 $a_1 - Bt_2$ 大小的数据需要传输, 这些数据都会以 $\frac{B}{2}$ 的速率传输, 因此第一条流的 FCT 为 $t_2 + \frac{a_1 - Bt_2}{\frac{B}{2}} = \frac{2a_1 - Bt_2}{B}$.
- 对于第二条流的 FCT, 仍可以类似看成两条流数据全部传输完成的时间即为 FCT, 但是需要注意减去开始时间 t_2, 也就是 $\frac{a_1+a_2}{B} - t_2$. 同样也可以分作两段（第一条流结束前, 第一条流完成后）分别计算, 结果是一样的.

Problem G - 下载时间显示器

总结:

$$\begin{cases} F_1 = \frac{a_1}{B}, F_2 = \frac{a_2}{B} & t_2 \geq \frac{a_1}{B} \\ F_1 = \frac{a_1+a_2}{B}, F_2 = \frac{2a_2}{B} & a_1 \geq Bt_2 + a_2 \\ F_1 = \frac{2a_1-Bt_2}{B}, F_2 = \frac{a_1+a_2}{B} - t_2 & \text{else} \end{cases}$$

Problem H - 完蛋, 我被房产包围了

题目大意

给定一系列处理存房和卖房事件, 要求按顺序处理这些事件, 并利用三个助理进行卖房, 使得获得的利润最大. 存房事件要求把一个房存入任意一个助理名下, 卖房事件会让每个助理自己选择一套房将其卖出并获得利润. 每个助理都有独特的自主选房策略和利润回报方式.

tag: 费用流, dp

Problem H - 完蛋，我被房产包围了 - 费用流做法

- 考虑每个房子的去处：被小红卖出，被小绿卖出，被小蓝卖出，未被卖出．在第一类预言出现时，我们贪心地考虑，则有以下策略：

Problem H - 完蛋，我被房产包围了 - 费用流做法

- 考虑每个房子的去处：被小红卖出，被小绿卖出，被小蓝卖出，未被卖出．在第一类预言出现时，我们贪心地考虑，则有以下策略：
  1. 如果我们希望这套房子**被小红卖出**，则我们把这套房子储存在**小红**手里；

Problem H - 完蛋，我被房产包围了 - 费用流做法

- 考虑每个房子的去处：被小红卖出，被小绿卖出，被小蓝卖出，未被卖出．在第一类预言出现时，我们贪心地考虑，则有以下策略：
  1. 如果我们希望这套房子**被小红卖出**，则我们把这套房子储存在**小红**手里；
  2. 如果我们希望这套房子**被小绿卖出**，则我们把这套房子储存在**小绿**手里；

Problem H - 完蛋，我被房产包围了 - 费用流做法

- 考虑每个房子的去处：被小红卖出，被小绿卖出，被小蓝卖出，未被卖出．在第一类预言出现时，我们贪心地考虑，则有以下策略：
  1. 如果我们希望这套房子**被小红卖出**，则我们把这套房子储存在**小红**手里；
  2. 如果我们希望这套房子**被小绿卖出**，则我们把这套房子储存在**小绿**手里；
  3. 如果我们希望这套房子**被小蓝卖出**，则我们把这套房子储存在**小蓝**手里；

Problem H - 完蛋，我被房产包围了 - 费用流做法

- 考虑每个房子的去处：被小红卖出，被小绿卖出，被小蓝卖出，未被卖出．在第一类预言出现时，我们贪心地考虑，则有以下策略：
  1. 如果我们希望这套房子**被小红卖出**，则我们把这套房子储存在**小红**手里；
  2. 如果我们希望这套房子**被小绿卖出**，则我们把这套房子储存在**小绿**手里；
  3. 如果我们希望这套房子**被小蓝卖出**，则我们把这套房子储存在**小蓝**手里；
  4. 如果我们希望这套房子**不被卖出**，则我们把这套房子储存在**小绿**手里．

Problem H - 完蛋，我被房产包围了 - 费用流做法

即对于小红和小蓝来说，我们只把需要他们卖出的房子交给他们；对于小绿来说，我们把需要他卖出的房子和完全放弃卖出的房子交给他. 由于小绿卖房时优先出售价值最高的房子，因此能获得最大利润的方案一定能通过上述策略构造出来.
通过上述策略，我们可以把三个助理卖房时优先出售某种房子这一限制解决掉，将问题等价地转换为：出现第一类预言时可以指定一个助理销售房子，也可以放弃卖出这套房子，且指定某助理销售时需要保证该助理能够通过买房狂潮销售所有指定给他的房子.

Problem H - 完蛋，我被房产包围了 - 费用流做法

对于转化后的问题，是一个经典的最小费用最大流模型，以下提供一种建图策略:

1. 建立源点 s 和汇点 t .

Problem H - 完蛋, 我被房产包围了 - 费用流做法

对于转化后的问题, 是一个经典的最小费用最大流模型, 以下提供一种建图策略:

1. 建立源点 s 和汇点 t .
2. 设第 i 条预言的事发生在时间点 i , 我们在时间点 i 对助理 j 建立一个对应的点 (i,j) , 即此步骤共建立 $3n$ 个点. (i,j) 向 $(i+1,j)$ 连边: 流量上限为 ∞ , 费用为 0 .

Problem H - 完蛋，我被房产包围了 - 费用流做法

对于转化后的问题，是一个经典的最小费用最大流模型，以下提供一种建图策略：

1. 建立源点 s 和汇点 t .
2. 设第 i 条预言的事发生在时间点 i ，我们在时间点 i 对助理 j 建立一个对应的点 (i,j) ，即此步骤共建立 $3n$ 个点. (i,j) 向 $(i+1,j)$ 连边：流量上限为 ∞ ，费用为 0 .
3. 对于事件发生在时间点 i 的第一类预言，新建点 id . s 向 id 连边：流量上限为 1 ，费用为 0 . id 向 $(i,1)$ 连边：流量上限为 1 ，费用为 1 ; id 向 $(i,2)$ 连边：流量上限为 1 ，费用为 $\lceil \frac{a_i}{10} \rceil$; id 向 $(i,3)$ 连边：流量上限为 1 ，费用为 0 ; id 向 t 连边：流量上限为 1 ，费用为 a_i .

Problem H - 完蛋，我被房产包围了 - 费用流做法

对于转化后的问题，是一个经典的最小费用最大流模型，以下提供一种建图策略:

1. 建立源点 s 和汇点 t .
2. 设第 i 条预言的事发生在时间点 i ，我们在时间点 i 对助理 j 建立一个对应的点 (i,j) ，即此步骤共建立 $3n$ 个点. (i,j) 向 $(i+1,j)$ 连边: 流量上限为 ∞ ，费用为 0 .
3. 对于事件发生在时间点 i 的第一类预言，新建点 id . s 向 id 连边: 流量上限为 1 ，费用为 0 . id 向 $(i,1)$ 连边: 流量上限为 1 ，费用为 1 ; id 向 $(i,2)$ 连边: 流量上限为 1 ，费用为 $\lceil \frac{a_i}{10} \rceil$; id 向 $(i,3)$ 连边: 流量上限为 1 ，费用为 0 ; id 向 t 连边: 流量上限为 1 ，费用为 a_i .
4. 对于事件发生在时间点 i 的第二类预言: $(i,1)$ 向 t 连边，流量上限为 1 ，费用为 0 ; $(i,2)$ 向 t 连边，流量上限为 1 ，费用为 0 ; $(i,3)$ 向 t 连边，流量上限为 1 ，费用为 0 .

Problem H - 完蛋，我被房产包围了 - 费用流做法

对于转化后的问题，是一个经典的最小费用最大流模型，以下提供一种建图策略:

1. 建立源点 s 和汇点 t .
2. 设第 i 条预言的事发生在时间点 i ，我们在时间点 i 对助理 j 建立一个对应的点 (i,j) ，即此步骤共建立 $3n$ 个点. (i,j) 向 $(i+1,j)$ 连边: 流量上限为 ∞ ，费用为 0 .
3. 对于事件发生在时间点 i 的第一类预言，新建点 id . s 向 id 连边: 流量上限为 1 ，费用为 0 . id 向 $(i,1)$ 连边: 流量上限为 1 ，费用为 1 ; id 向 $(i,2)$ 连边: 流量上限为 1 ，费用为 $\lceil \frac{a_i}{10} \rceil$; id 向 $(i,3)$ 连边: 流量上限为 1 ，费用为 0 ; id 向 t 连边: 流量上限为 1 ，费用为 a_i .
4. 对于事件发生在时间点 i 的第二类预言: $(i,1)$ 向 t 连边，流量上限为 1 ，费用为 0 ; $(i,2)$ 向 t 连边，流量上限为 1 ，费用为 0 ; $(i,3)$ 向 t 连边，流量上限为 1 ，费用为 0 .
5. 最终答案为 a_i 之和减去最小费用最大流的最小费用.

Problem H - 完蛋, 我被房产包围了 - 费用流做法

单组数据空间复杂度 $O(n)$ ，时间复杂度上限为 $O(n^3)$ ，实际上由于图的特殊性时间复杂度低于该上限.

Problem H - 完蛋，我被房产包围了 - 动态规划做法

- 同样使用一定的贪心策略将问题转化为：出现第一类预言时可以指定一个助理销售房子，也可以放弃卖出这套房子，且指定某助理销售时需要保证该助理能够通过买房狂潮销售所有指定给他的房子.

Problem H - 完蛋，我被房产包围了 - 动态规划做法

- 同样使用一定的贪心策略将问题转化为：出现第一类预言时可以指定一个助理销售房子，也可以放弃卖出这套房子，且指定某助理销售时需要保证该助理能够通过买房狂潮销售所有指定给他的房子.
- 首先考虑这样一个动态规划：

Problem H - 完蛋，我被房产包围了 - 动态规划做法

- 同样使用一定的贪心策略将问题转化为：出现第一类预言时可以指定一个助理销售房子，也可以放弃卖出这套房子，且指定某助理销售时需要保证该助理能够通过买房狂潮销售所有指定给他的房子.
- 首先考虑这样一个动态规划:
  - 定义 dp_{i,j_1,j_2,j_3} 表示考虑前 i 个预言的事件，三个助理还需要销售的房子数量分别为 j_1 j_2 j_3 时的最大利润，答案即 $dp_{n,0,0,0}$.

Problem H - 完蛋, 我被房产包围了 - 动态规划做法

- 若第 i 个预言为第一类预言, 则有转移:

Problem H - 完蛋, 我被房产包围了 - 动态规划做法

- 若第 i 个预言为第一类预言, 则有转移:
  - $dp_{i,j_1,j_2,j_3} = \max\{dp_{i-1,j_1,j_2,j_3}, dp_{i-1,j_1-1,j_2,j_3} + a_i - 1, dp_{i-1,j_1,j_2-1,j_3} + a_i - \lceil \frac{a_i}{10} \rceil, dp_{i-1,j_1,j_2,j_3-1} + a_i\}$

Problem H - 完蛋, 我被房产包围了 - 动态规划做法

- 若第 i 个预言为第一类预言, 则有转移:
  - $dp_{i,j_1,j_2,j_3} = \max\{dp_{i-1,j_1,j_2,j_3}, dp_{i-1,j_1-1,j_2,j_3} + a_i - 1, dp_{i-1,j_1,j_2-1,j_3} + a_i - \lceil \frac{a_i}{10} \rceil, dp_{i-1,j_1,j_2,j_3-1} + a_i\}$
- 若第 i 个预言为第二类预言, 则有转移:

Problem H - 完蛋, 我被房产包围了 - 动态规划做法

- 若第 i 个预言为第一类预言, 则有转移:
  - $dp_{i,j_1,j_2,j_3} = \max\{dp_{i-1,j_1,j_2,j_3}, dp_{i-1,j_1-1,j_2,j_3} + a_i - 1, dp_{i-1,j_1,j_2-1,j_3} + a_i - \lceil \frac{a_i}{10} \rceil, dp_{i-1,j_1,j_2,j_3-1} + a_i\}$
- 若第 i 个预言为第二类预言, 则有转移:
  - $dp_{i,\max\{0,j_1-1\},\max\{0,j_2-1\},\max\{0,j_3-1\}} \leftarrow dp_{i-1,j_1,j_2,j_3}$

Problem H - 完蛋，我被房产包围了 - 动态规划做法

- 若第 i 个预言为第一类预言，则有转移:
  - $dp_{i,j_1,j_2,j_3} = \max\{dp_{i-1,j_1,j_2,j_3}, dp_{i-1,j_1-1,j_2,j_3} + a_i - 1, dp_{i-1,j_1,j_2-1,j_3} + a_i - \lceil \frac{a_i}{10} \rceil, dp_{i-1,j_1,j_2,j_3-1} + a_i\}$
- 若第 i 个预言为第二类预言，则有转移:
  - $dp_{i,\max\{0,j_1-1\},\max\{0,j_2-1\},\max\{0,j_3-1\}} \leftarrow dp_{i-1,j_1,j_2,j_3}$
- 上述动态规划的时间复杂度为 $O(n^4)$.

Problem H - 完蛋, 我被房产包围了 - 动态规划做法

- 考虑优化, 将小红和小蓝需要销售的房子合并在一起考虑. 若第 i 个预言是第一类预言, 则在指定这两人销售时, 直接将 a_i 作为利润计入贡献, 即预先假设由小蓝销售. 销售时考虑两人卖一套和两套两种情况. 若卖一套, 则交给小蓝销售, 此时利润不会减少; 若卖两套, 则小红和小蓝各卖一套, 此时与假设不符合, 利润将减少 1 .

Problem H - 完蛋, 我被房产包围了 - 动态规划做法

- 考虑优化, 将小红和小蓝需要销售的房子合并在一起考虑. 若第 i 个预言是第一类预言, 则在指定这两人销售时, 直接将 a_i 作为利润计入贡献, 即预先假设由小蓝销售. 销售时考虑两人卖一套和两套两种情况. 若卖一套, 则交给小蓝销售, 此时利润不会减少; 若卖两套, 则小红和小蓝各卖一套, 此时与假设不符合, 利润将减少 1 .
- 以下为优化后的完整动态规划过程:

Problem H - 完蛋, 我被房产包围了 - 动态规划做法

- 考虑优化, 将小红和小蓝需要销售的房子合并在一起考虑. 若第 i 个预言是第一类预言, 则在指定这两人销售时, 直接将 a_i 作为利润计入贡献, 即预先假设由小蓝销售. 销售时考虑两人卖一套和两套两种情况. 若卖一套, 则交给小蓝销售, 此时利润不会减少; 若卖两套, 则小红和小蓝各卖一套, 此时与假设不符合, 利润将减少 1 .
- 以下为优化后的完整动态规划过程:
  - 定义 dp_{i,j_1,j_2} 表示考虑前 i 个预言的事件, 小绿还需要销售的房子数量分别为 j_1 , 小红和小蓝还需要销售的房子数量为 j_2 时的最大利润, 答案即 $dp_{n,0,0}$.

Problem H - 完蛋, 我被房产包围了 - 动态规划做法

- 若第 i 个预言为第一类预言, 则有转移:

Problem H - 完蛋, 我被房产包围了 - 动态规划做法

- 若第 i 个预言为第一类预言, 则有转移:
  - $dp_{i,j_1,j_2} = \max\{dp_{i-1,j_1,j_2}, dp_{i-1,j_1,j_2-1} + a_i, dp_{i-1,j_1-1,j_2} + a_i - \lceil \frac{a_i}{10} \rceil\}$

Problem H - 完蛋，我被房产包围了 - 动态规划做法

- 若第 i 个预言为第一类预言，则有转移:
  - $dp_{i,j_1,j_2} = \max\{dp_{i-1,j_1,j_2}, dp_{i-1,j_1,j_2-1} + a_i, dp_{i-1,j_1-1,j_2} + a_i - \lceil \frac{a_i}{10} \rceil\}$
- 若第 i 个预言为第二类预言，则有转移:

Problem H - 完蛋，我被房产包围了 - 动态规划做法

- 若第 i 个预言为第一类预言，则有转移:
  - $dp_{i,j_1,j_2} = \max\{dp_{i-1,j_1,j_2}, dp_{i-1,j_1,j_2-1} + a_i, dp_{i-1,j_1-1,j_2} + a_i - \lceil \frac{a_i}{10} \rceil\}$
- 若第 i 个预言为第二类预言，则有转移:
  - $dp_{i,\max\{0,j_1-1\},\max\{0,j_2-1\}} \leftarrow dp_{i-1,j_1,j_2}$
  - $dp_{i,\max\{0,j_1-1\},\max\{0,j_2-2\}} \leftarrow dp_{i-1,j_1,j_2} - 1$

Problem H - 完蛋, 我被房产包围了 - 动态规划做法

- 若第 i 个预言为第一类预言, 则有转移:
  - $dp_{i,j_1,j_2} = \max\{dp_{i-1,j_1,j_2}, dp_{i-1,j_1,j_2-1} + a_i, dp_{i-1,j_1-1,j_2} + a_i - \lceil \frac{a_i}{10} \rceil\}$
- 若第 i 个预言为第二类预言, 则有转移:
  - $dp_{i,\max\{0,j_1-1\},\max\{0,j_2-1\}} \leftarrow dp_{i-1,j_1,j_2}$
  - $dp_{i,\max\{0,j_1-1\},\max\{0,j_2-2\}} \leftarrow dp_{i-1,j_1,j_2} - 1$
- 单组数据时间复杂度为 $O(n^3)$, 空间复杂度可通过滚动数组技巧优化到 $O(n^2)$.

Problem I - 整数反应

题目大意

一列 n 个数, 每个数有两种颜色. 这些数按从左到右顺序依次进入一个多重集合 S_1, 如果 S_1 中有与其不同颜色的数, 就选一个进行反应, 将这两个数的和放入 S_2 集合中, 这两个数均消失; 否则添加到 S_1 集合中, 不发生反应. 求 S_2 中最小元素的可能最大值.

tag: 二分答案, 贪心

Problem I - 整数反应

- 由于至少存在一次反应, S_2 中最小元素一定存在最大值, 因此考虑二分答案. 则问题转化成一个判定问题, 即: 是否存在一种反应的方案, 使得 S_2 中的最小元素不小于 x.

Problem I - 整数反应

- 由于至少存在一次反应, S_2 中最小元素一定存在最大值, 因此考虑二分答案. 则问题转化成一个判定问题, 即: 是否存在一种反应的方案, 使得 S_2 中的最小元素不小于 x.
- 考虑贪心地进行反应. 维护多重集合 S_1, 容易知道每时每刻 S_1 中元素颜色均相同, 每当枚举到与 S_1 中元素颜色不同的元素 y 时, 选择大于等于 $x-y$ 的最小元素与其反应, 如果不存在这样的元素, 则判定不存在方案. 如果成功处理到最后, 则说明存在选择方案.

Problem I - 整数反应

- 这样贪心的正确性在于：由于要求 S_2 中最小的元素不小于 x，那么一定需要选择大于等于 $x-y$ 的元素与其反应，满足选择性质；如果选择了大于等于 $x-y$ 的元素，我们总可以选择最小的一个，使后续较小的 y 可以匹配到较大的元素，如果交换这两次选择，后续较小的 y 将可能无法匹配，因此选择最小的一个具有优化子结构.
- 时间复杂度：$\mathcal{O}(n \log n \log \max\{a_i\})$，空间复杂度：$\mathcal{O}(n)$.

Problem J - 骨牌覆盖

题目大意

给定一个 $n \times m$ 的网格，每个格子有 $a_{i,j}$ 的权值，问用一边长为 1，另一边任意的矩形去覆盖，满足矩形不四联通的情况下覆盖的最大权值和.

tag：dp

Problem J - 骨牌覆盖

- 使用插头 dp.

Problem J - 骨牌覆盖

- 使用插头 dp.
- 设 $f_{i,j,S}$ 表示从上到下, 从左到右 dp 到 (i,j) 位置, 用 S 记录每列的是否被覆盖, 此时的最大权值和.

Problem J - 骨牌覆盖

- 使用插头 dp.
- 设 $f_{i,j,S}$ 表示从上到下, 从左到右 dp 到 (i,j) 位置, 用 S 记录每列的是否被覆盖, 此时的最大权值和.
- 转移考虑是否覆盖 $(i,j+1)$ 位置, 不覆盖直接转移, 覆盖考虑我们不能在相邻两行出现形如 L 的形状, 所以 S 多记录一位表示 $(i-1,j)$ 是否被覆盖, 然后转移即可.

Problem J - 骨牌覆盖

- 使用插头 dp.
- 设 $f_{i,j,S}$ 表示从上到下, 从左到右 dp 到 (i, j) 位置, 用 S 记录每列的是否被覆盖, 此时的最大权值和.
- 转移考虑是否覆盖 $(i, j+1)$ 位置, 不覆盖直接转移, 覆盖考虑我们不能在相邻两行出现形如 L 的形状, 所以 S 多记录一位表示 $(i-1, j)$ 是否被覆盖, 然后转移即可.
- 使用滚动数组优化, 空间复杂度 $O(m2^m)$, 时间复杂度 $O(nm2^m)$.

Problem K - 删数游戏

题目大意

删数游戏，每次可以删除一个最大值，并加入任意一个小于最大值的前缀的所有数（可以为空），问先后手谁赢.
tag：博弈

Problem K - 删数游戏

- 结论: 统计最大值数量的奇偶性, 如果是奇, 先手胜, 否则后手胜, 复杂度 $O(n)$.

Problem K - 删数游戏

- 结论: 统计最大值数量的奇偶性, 如果是奇, 先手胜, 否则后手胜, 复杂度 $O(n)$.
- 证明: 按最大值从小到大归纳.

Problem K - 删数游戏

- 结论: 统计最大值数量的奇偶性, 如果是奇, 先手胜, 否则后手胜, 复杂度 $O(n)$.
- 证明: 按最大值从小到大归纳.
  - 对 $\max = 1$, 结论显然成立.

Problem K - 删数游戏

- 结论: 统计最大值数量的奇偶性, 如果是奇, 先手胜, 否则后手胜, 复杂度 $O(n)$.
- 证明: 按最大值从小到大归纳.
  - 对 $\max = 1$, 结论显然成立.
  - 若对 $\max = 1, 2, \ldots, k$ 成立, 则对 $\max = k + 1$:

Problem K - 删数游戏

- 结论: 统计最大值数量的奇偶性, 如果是奇, 先手胜, 否则后手胜, 复杂度 $O(n)$.
- 证明: 按最大值从小到大归纳.
  - 对 $\max = 1$, 结论显然成立.
  - 若对 $\max = 1, 2, \ldots, k$ 成立, 则对 $\max = k + 1$:
    - 如果 $k + 1$ 有奇数个, 那么无论之前操作如何, 最后一次操作 $k + 1$ 先手可以决定次大值的奇偶性为偶, 那么由归纳假设, 先手胜.

Problem K - 删数游戏

- 结论: 统计最大值数量的奇偶性, 如果是奇, 先手胜, 否则后手胜, 复杂度 $O(n)$.
- 证明: 按最大值从小到大归纳.
  - 对 $\max = 1$, 结论显然成立.
  - 若对 $\max = 1, 2, \ldots, k$ 成立, 则对 $\max = k + 1$:
    - 如果 $k + 1$ 有奇数个, 那么无论之前操作如何, 最后一次操作 $k + 1$ 先手可以决定次大值的奇偶性为偶, 那么由归纳假设, 先手胜.
    - 同理如果 $k + 1$ 为偶数个, 后手胜.

Problem K - 删数游戏

- 结论: 统计最大值数量的奇偶性, 如果是奇, 先手胜, 否则后手胜, 复杂度 $O(n)$.
- 证明: 按最大值从小到大归纳.
  - 对 $\max = 1$, 结论显然成立.
  - 若对 $\max = 1, 2, \ldots, k$ 成立, 则对 $\max = k + 1$:
    - 如果 $k + 1$ 有奇数个, 那么无论之前操作如何, 最后一次操作 $k + 1$ 先手可以决定次大值的奇偶性为偶, 那么由归纳假设, 先手胜.
    - 同理如果 $k + 1$ 为偶数个, 后手胜.
  - 则结论对 $\max = k + 1$ 成立, 证毕.