Notes

A. Programming Contest

签.

直接计算起始年份和当前年份有多少停办即可.

B. Base Station Construction

线段树优化 dp (单调队列也行). 要求给定区间内选点, 求最小花费.

用 f_i 表示 i 选择的并且向前满足要求的最小代价.
这样只需要计算 f_{n+1} 即可, 如果将 a_{n+1} 设置为 0.

考虑每一个点 i 对应的 p_i 满足 $[p_i, i]$ 中一定要选择点的最大的 p_i.
那对于每一个区间 $[l, r]$ 而言, 只需要将 $[r+1, n+1]$ 的点与 l 取 $\max$ 即可, 这可以用 BIT 实现.

接着 dp 转移方程就变成了

$$f_i = a_i + \min_{p_i \leqslant j \leqslant i-1} a_j,$$

这个可以在线段树上查询, 注意一下 $p_i = 0$ 的情形, 因为线段树的建立不一定有 0 这个点.

C. Trading

计算? 在 n 家商店之间倒买倒卖, 每一家商店有操作上限, 求最大利润.

将价格排序, 一定是在价格小的一半买, 价格大的一半买, 然后再中间某一家或者相邻两家碰头.

直接算就行.

D. New Houses

贪心. n 个人 m 间房, 每个人在有邻居和没有邻居的不同情况有不同的喜好值, 求最大喜好值.

首先可以考虑所有人相邻而住的喜好值之和, 接着每多一间房子, 就可以使一个人从有邻居的状态变成没有邻居的状态.
将所有的 $b_i - a_i$ 排序, 优先差值大的.

注意一个人的情形, 这个人是没有邻居的.
注意最后两个人的情形, 多一间屋子这两个人是同时变成没有邻居的状态, 可以单独计算.

E. New but Nostalgic Problem

trie. 在 n 个字符串中选择 k 个, 求最小的 k 个字符串中两两 lcp 的最大值.

首先将 n 个字符串放在 trie 上. 接着就可以在 trie 上贪答案.
假设目前已经确定的答案是 s, 接着要确定下一位, 为了让字典序尽可能小, 就需要从 a 到 z 依次枚举.

假设枚举到字符 d, 考虑从当前trie节点上，字符 $c(c \leq d)$ 指向的节点对应的字符串数量加上剩余边数大于等于 k 时，则往该方向下走，并且将 k 减掉字符 $c(c < d)$ 指向的节点对应的字符串数量，同时减掉以当前节点为结尾的字符串个数。

当 k 的数量小于等于当前节点延伸出去边的数量时，则停止遍历。

最终得到的字符串即为答案。

F. Traveling in Cells

线段树二分. 1 到 n 有各自的权值和颜色, 有三种操作:

1. 修改单点的权值,
2. 修改单点的颜色,
3. 在只能走给定颜色的前提下, 给定起点, 求可到达位置的权值和.

线段树维护区间的值的和以及所含颜色的数量. 针对每一个操作而言:

1. 单点修改维护区间权值和, 需要 push_up.
2. 单点修改维护区间每种颜色各种的数量, 修改的过程就是在线段树上进过的点删除之前的颜色, 加上之后的颜色, 如果是 push_up 合并两个儿子的颜色可能会时间复杂度不对.
3. 容易发现可达区间是连续的, 可以在线段树上二分, 在二分前记录哪些颜色可达, 然后在线段树上二分可达区间 $[l, r]$, 输出区间 $[l, r]$ 的权值和即可.

I. Path Planning

贪心加创造性算法? 求所有从左上角到右下角的路径 mex 的最大值.

注意值是 0 到 $n * m - 1$ 并且两两不重复, 以及路径是只能向右或向下.

考虑贪心, 依次考虑 $0, 1, 2, \cdots, n * m - 1$ 能不能选.

注意到每一向左下斜的一列点只能经过一个, 再考虑 i (对应的格子为 (x, y)) 能不能选, 这意味着 $0, 1, \cdots, i - 1$ 已经是可以走的.
如果用一个 vector 维护所有的 $x + y$, 接着就考虑 $x + y$ 的上一个点和下一个点, 构成的矩阵是否能包括当前点, 如果可以意味着 i 可, 否则不可选.

可以在初始时在 vector 中塞入 0 和 $n + m$ 保证能二分出上界的下界.

J. X Equals Y

分类讨论. 给的数 x, y 以及上界 A, B, 求两个进制 $a \in [2, A], b \in [2, B]$ 使得 x 在 a 进制和 y 在 b 进制下表示是一样的.

首先 $x = y$ 可以直接取 $a = 2, b = 2$.

接着考虑进制表示超过 2 位的情形.
这时 $a \leqslant \sqrt{x}, b \leqslant \sqrt{y}$.
可以枚举 a, 强行计算进制, 用 map 存储, 再枚举 b, 依旧是强行计算进制, 然后在 map 中查找.

最后是进制表示为 2 位的情形, 这里不妨设 $x > y$.
可以设 $x = s + at, y = s + bt$, 作差可以发现 $(a-b)t = x - y$, 也就是可以在 $t \mid (x-y)$ 且 $t^2 \leqslant \max\{x, y\}$ 中枚举高位 t. 这时 $a = b + \frac{x-y}{t} := b + z$.
可以考虑一些不等式, 例如:

$$\begin{cases} t \in [1, a-1] \cap [1, b-1] \Rightarrow t + 1 \leqslant a \text{ and } t + 1 + z \leqslant a, \\ x - ta \in [0, a-1] \text{ and } y - tb \in [0, b-1] \Rightarrow \dfrac{x+1}{t+1} \leqslant a \text{ and } \dfrac{y+1}{t+1} + z \leqslant a, \\ a \in [2, A] \text{ and } b \in [2, B] \Rightarrow a \leqslant \dfrac{x}{t} \text{ and } a \leqslant \dfrac{y}{t} + z \text{ and } B + z. \end{cases}$$

如果 a 有界, 算出 b 即为答案.

K. Peg Solitaire

dfs. 6×6 的棋盘上下跳棋, 问无法操作后最少可以剩几个.

纯纯 dfs 暴搜, 可以加上状压减少空间花费.

M. Peg Solitaire

套着计算几何外壳的 dp? 将凸多边形沿两个顶点的连线切成两部分, 求两个小凸包的直径的平方和的最小值.

通过数据范围可以发现, 可以枚举切开的两个顶点, 接着就需要快速计算左右两块的直径.

考虑如下 dp 状态, 设 $dp_{i,j}$ 表示凸包沿顶点 v_i, v_j (有方向) 右侧部分的直径平方, 初值就是 $f_{i,i+1} = (dist(v_i, v_{i+1}))^2$.
考虑转移方程如下

$$f_{i,j} = \min\{f_{i+1,j}, f_{i,j-1}, (dist(v_i, v_j))^2\},$$

这时因为凸包的直径一定是顶点的连线, 所以上述转移方程是对的.

接着就只需要枚举 i, j 使得 v_i, v_j 连线不是凸包的边, 答案与 $f_{i,j} + f_{j,i}$ 取 $\min$ 即可.