IOI2014 国家集训队第一次作业
试题泛做部分题解

东营市胜利第一中学 王子昱

Contents

1 2001 ∼ 2005

1.1 01A. Airport Configuration

1.1.1 题意

有若干机场布局方案，每个方案可以用两个 1 到 N 的排列 (P, Q) 描述。计算每个方案的客流指数，并将它们升序输出。

方案 (P, Q) 的客流指数为

$$\sum_{i=1}^{n}\sum_{j=1}^{n}(|i-j|+1)G_{ij}$$

，其中 G 为给定矩阵。

$N < 25$，方案数 $K \leq 20$。

1.1.2 题解

按照定义计算每个方案的客流指数，并排序输出。算法时间复杂度 $O(K(N^2 + \log K))$。

1.2 01C. Crossword Puzzle

1.2.1 题意

一个填字游戏在 10×10 的网格上进行，网格中有 N 个槽可以放入单词。每个槽由其起始位置和延续方向（向右或向下）描述，槽的长度没有限制。游戏给定了 $N + 1$ 个两两互异的候选单词，其中有一个多余。

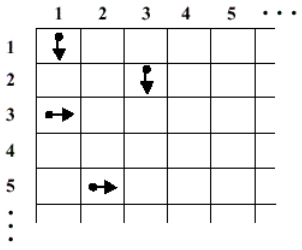

Figure 1: Corner of Example Puzzle

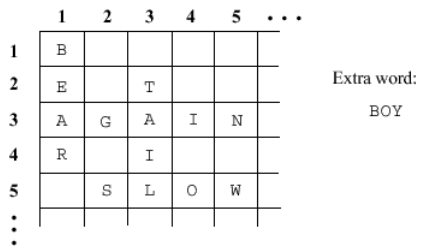

Figure 2: Example Solution

一个槽上的单词是指从其起始位置出发，沿其指定方向延伸得到的最长字符串。在合法的方案中，每个槽上都有一个单词，每个单词至多出现在一个槽中。为了避免混淆，每个槽的起始位置之前（延伸方向的反方向）一格必须为空。单词之间不能冲突。

确定给定的单词中多余的一个。

1.2.2 题解

注意到这一问题至少是 NP 完全[1]的，所以我们考虑对朴素的搜索进行优化。

固定槽的顺序，对单词的安排顺序进行搜索。每尝试放下一个单词时，我们检查它是否与之前的单词冲突，并检查是否存在某个槽使得此单词被放下后，其起始位置的前一位置被占用。

[1]这里列出了本题的简化版。

除此之外，对未占用的槽，如果从它的起始位置出发已经可以连成一个字符串，我们检查在未被使用的单词中，是否存在一个单词使得该串为其前缀。这一检查可以用 Trie 树完成。

为了使上述检查尽量能在开始时约束搜索树的大小，我们将所有的槽按行列坐标排序。此外，对单词按长度排序也有很好的效果。

以上优化足以通过本题的测试数据。

1.3 02H. Silly Sort

1.3.1 题意

给定一个整数序列，用最小的代价将它排成升序。排序中，每次操作可以交换两个数的位置，交换代价为这两个数之和。

序列中所有数互异；序列长度 $N \leq 1000$，序列中最大值 $M \leq 1000$。

1.3.2 题解

考察原序列到排序后序列的置换。对于其中的每个轮换，我们可以用其中的最小元 a 与其余所有数交换，也可以将整个序列的最小元 b 与 a 交换，之后用 b 与轮换中的数交换，并最终交换 a 和 b。对每个轮换，选取其中更优的一种策略完成排序。易证不存在更优的策略。

算法复杂度 $O(N+M)$。

1.4 05B. Simplified GSM Network

1.4.1 题意

手机在移动过程中总会使用距离最近的基站。

给定平面上 N 个基站，M 座城市的坐标，以及 K 条（直线段）道路。找出从 s 到 t 的一条路径，使得手机沿该路径移动时切换基站的次数最少。

$N, M < 50, K < 2500$；输入保证道路上的任意点不会在三座基站的服务范围内，任意一座城市不会在两座基站的服务范围内。

1.4.2 题解

计算在每条路上切换基站的次数，Floyd 算法求出最短路。

计算一条线段上的基站切换次数可以用扫描线在 $O(N)$ 时间内完成，这里不再赘述。算法时间复杂度为 $O(KM^2 + N^3)$。

1.5 05C. The Traveling Judges Problem

1.5.1 题意

给定一个无向正权图和点集的子集 S，求连接 S 的最小权子图。

$|V|, |S| \leq 20$。

1.5.2 题解

动态规划。设 $f(S)$ 为连接 S 的最小权子图，$g(S,k)$ 为连接 k 和 S 中任意一点的最短路。于是

$$\begin{aligned} g(S,k) &= \min_{i=\min(S)}(g(S-\{i\},k),dis(i,k)) \\ f(S) &= \min_{i\in S}(f(S-\{i\})+g(S-\{i\},i)) \end{aligned}$$

算法复杂度 $O(|V|2^{|V|})$。

1.6 05E. Lots of Sunlight

1.6.1 题意

给定排在直线上的 N 栋楼，每栋楼可以看做由若干全等矩形上下堆叠而成。Q 组询问，每次给定一栋楼上的某一层，求其被阳光直射的时间。

$N, Q \leq 1000$。

1.6.2 题解

对每个询问，$O(N)$ 找到日出时最后挡住给定楼层的楼，以及日落前最先挡住给定楼层的楼，计算答案。

1.7 05F. Crossing Streets

1.7.1 题意

给定 N 条平行于坐标轴的线段以及两点 S T，求一条连接 S 与 T 的路径使其穿过最少的线段。路径不能穿过线段的交点。

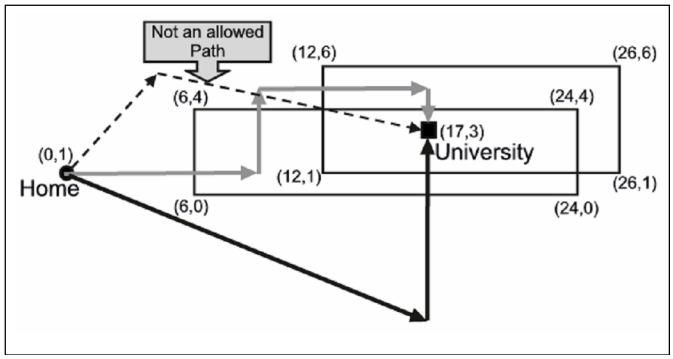

Figure 1: 合法路径的实例。

$N \leq 500$。

1.7.2 题解

离散化后坐标系中只剩下 $O(N^2)$ 个区域。枚举答案，BFS 验证。

假设当前枚举到的答案为 i，计算从 S 出发穿过 i 条路能到达的区域集合，判断 T 所在区域是否被访问。用队列维护可达的区域集合，当 i 增一时，对于队列中的每个区域，枚举其沿四个方向

穿过一条线段可以到达的区域，如果未被访问则将它加入新队列，最后 BFS 计算与队列中新区域联通的范围。

由于每个区域只被访问一次，算法复杂度 $O(N^2)$。

1.8 05H. The Great Wall Game

1.8.1 题意

在 $N \times N$ 的棋盘上有 N 枚棋子，每一步你可以将一枚棋子移到任意一个相邻的方格上，求使所有棋子共线（共行，共列或共对角线）最少需要移动的步数。

$N \leq 15$。

1.8.2 题解

考虑最终棋子在同一行的情况。显然我们可以将横向和纵向的移动分开考虑。纵向移动应使所有棋子都移到纵坐标的中位数所对应的行上；横向移动中，横坐标第 i 小的棋子应移到第 i 列上。

最终棋子共列的情况同理；共对角线的情况可以用最小权匹配解决。

算法复杂度 $O(N^3)$。

1.9 05I. Workshops

1.9.1 题意

有 N 个讨论会和 M 个房间，每个讨论会必须在不同的房间中进行。所有会议从同一时间开始，在不同时间结束，参加人数不同；每个房间有不同的使用截止时间，在其中进行的会议必须在这一时间之前结束；每个房间最多可以容纳的人数不同。

选出尽量多的讨论会，使每个讨论会都可以分配到一个房间；如果有多解，选择参加人数之和最多的一个。

$N, M \leq 1000$。

1.9.2 题解

如果没有人数的限制，只需要按截止时间从前到后考虑每个房间，如果在剩下的会议中有可以分配到当前房间的，就将它分配到该房间。增加人数限制之后，仍然按截止时间考虑每个房间，将每个房间分配给可以分配的会议中，人数最多的一个即可。

为了证明该算法的正确性，我们可以递归应用以下结论。

定理 1.1. *设上述算法在某一输入上最先选择的房间为 r，对应会议为 m，则该输入的任意一个方案在引入 (r, m) [2] 并进行调整之后都不会更差。*

证明. 设某一方案 P 中选择的会议集合为 M，房间集合为 R。并设 P_i 表示房间（或会议）i 的人数，T_i 表示 i 的截止时间，pal_i 表示房间（或会议）i 对应的会议（或房间）。考虑以下几种情况。

- $r \notin R, m \notin M$：引入 (r, m) 显然得到更优方案。
- $r \notin R, m \in M$：将 pal_m 换成 r，得到的方案不会更差。

[2] 指在方案中将房间 r 匹配给会议 m，以下同。

- $r \in R, m \notin M$：根据算法描述 $P_{pal_r} \leq P_m$，所以将 pal_r 换为 m 不会得到更差的方案。
- $r \in R, m \in M, pal_r \neq m$：设 $r' = pal_m, m' = pal_r$。由于上述算法选择的第一个房间－会议对为 (r, m)，$T_{r'} \geq T_r$，且 $P_{m'} \leq P_m$。于是 $P_{r'} \geq P_m \geq P_{m'}$，$T_{r'} \geq T_r \geq T'_m$，交换 pal_r 和 $pal_{r'}$ 得到的方案仍然可行。

□

1.10 05J. Zones

1.10.1 题意

有 N 个服务塔，第 i 个服务塔覆盖了 A_i 名用户。有 M 个公共服务区，第 i 个公共服务区被服务塔 $S_{i1}..S_{ik_i}$ 覆盖，且包含 i 名用户。选择 K 个服务塔，使之覆盖最多的用户。

$N \leq 20$。

1.10.2 题解

枚举选择的服务塔集合，更新答案。时间复杂度 $O(MC_K^N)$。

2 2006 San Antonio

2.1 A. Low Cost Air Travel

2.1.1 题意

有 N 条航线，每条航线有固定的票价，并会按顺序经过若干城市。乘客可以从某条航线的起始城市出发，在中间的任意一个城市转机。

给定 M 座城市 $C_1..C_M$，计算一个转机方案从 C_1 出发，按顺序经过 $C_1..C_M$（中间可以经过其他城市），且花费最小。

$N, M \leq 20$，每条线路包含不超过 20 座城市。

2.1.2 题解

将原图中的点 v 拆成 $(v,0)..(v,M)$，点 (v,k) 表示从 C_1 出发，目前在点 v，已经经过了 C 中的 k 座城市的最小代价。在新图中加边求最短路。

时间复杂度 $O(V^2M^2)$。

2.2 B. Remember the A La Mode!

2.2.1 题意

有 N 种饼干和 M 种冰激凌，第 i 种饼干的数量为 A_i，第 i 种冰激凌的数量为 B_i。求出将饼干和冰激凌两两配对可以得到的最小总价值和最大总价值。将一片饼干 i 和一个冰激凌 j 搭配可以得到 C_{ij} 的价值。

$N, M \leq 50$。

2.2.2 题解

类似最大权匹配构图，费用流求解。注意变换边权使图中无负环。

2.3 D. Bipartite Numbers

2.3.1 题意

给定一个数 A，求其最小的倍数 B，满足 B 的十进制表示具有 “a...b...”（$a \neq b$）的形式。

$A < 10^5$；$T \leq 200$ 组数据捆绑测试。

2.3.2 题解

设 B 的十进制表示包含 l 个 a 和 m 个 b，于是

$$B = f(a, l+m) - f(a, m) + f(b, m) = f(a, l+m) + f(b-a, m)$$

，其中

$$f(x, y) = \frac{(10^y - 1)x}{9}$$

。所以题目等价于求出 (a, l, b, m) 使得

$$f(a, l+m) \equiv f(a-b, m) \ (mod\ A)$$

，且其对应的 B 最小。

由于 $f(c, x)$ 在模 A 意义下循环，问题必然有解，且 $l, m \leq A$。于是预处理 $f(x, y)$ 及 $g(x, y) = min(z : f(x, z) = y)$，枚举 $l+m$，a，b，如果有解继续枚举 l 更新答案。利用 g 的信息可以在 $O(1)$ 时间内判断给定的 $(l+m, a, b)$ 组合是否有解，所以算法复杂度为 $O(TA)$。

2.4 E. Bit Compressor

2.4.1 题意

考虑一种 01 串上的压缩算法：串中的 0 维持不变；原串中连续的 n 个 1 替换为 n 的二进制表示（如果替换后原串长度不变，替换不会发生）。

这一压缩算法显然会产生歧义。给定压缩后的串，原串长度以及原串中 1 的个数，判断是否存在对应原串，如果存在，判断原串是否唯一。

压缩后的串长 $M \leq 40$，原串长 $L \leq 2^{14}$。

2.4.2 题解

注意到原串中 0 的数量不超过 M，所以可以动态规划。$f(i, j, L)$ 表示压缩后串的第 i 到 M 位对应原串的长度为 L，含 0 个数为 j 的方案数，$O(M)$ 转移。算法复杂度 $O(M^2 \min(M, \log L)L)$。

2.5 G. Pilgrimage

2.5.1 题意

一些人在旅行。旅行过程中常需要用钱，所以他们维护了一笔公有资金。需要用钱时，他们从这笔钱中支出相应的数额；必要时他们会每人捐出相同数额的钱作为公用。

旅行中常有人离开或有人加入。有人离开时，他会从公用资金中得到“当前金额 / 人数”元，有人加入时他需要捐出“当前金额 / 人数”元。

上述四种事件均被记录。现给定记录的一个片段，已知公用资金的金额始终为整数，求记录片段开始时可能的人数。

记录数 $N \leq 50$；每次支出金额 $W \leq 2000$。

2.5.2 题解

设公用资金共有 $Ax + B$ 元，其中 x 为当前人数。开始时 A 取任意足够大的数，B 取一个合适的数使记录中的第一次加入或离开事件合法。

遍历记录获得对 x 取值的约束。记录中的支出和收入事件对应着对 A, B 的修改；有人加入或离开时，我们可以得知 B 被当前的 x 整除，由此可以得到开始时 x 值的取值限制。

复杂度 $O(N\sqrt{NW})$。

2.6 I. Degrees of Seperation

2.6.1 题意

给定无向图 $< V, E >$，求其中最短距离最大的点对。

点数 $N \leq 50$。

2.6.2 题解

Floyd。

2.7 J. Routing

2.7.1 题意

给定有向图 $< V, E >$ 和点 s, t，求点数最小的子图 G' 使得 G' 中同时存在从 s 到 t 和从 t 到 s 的路径。

$|V| \leq 100$。

2.7.2 题解

预处理 $W(u, v)$ 表示从 u 到 v 的最短路。

设 $d(u, v)$ 为同时存在 s 到 u 和 v 到 t 的路径的最小权子图，于是

$$d(u, v) = min(d(u', v') + W(u, u') + W(v', v))$$

（特殊处理 $u = v$ 等情况），初始时 $d(s, s) = 0$，$d(t, t)$ 即为所求。

由于表达式中的常数项恒正，我们可以用 Dijkstra 算法计算 d 值。复杂度 $O(V^4)$。

注意到

$$\begin{aligned} d(u, v) &= min(d(u', v') + W(u, u') + W(v', v)) \\ &= min(d(u', v) + W(v', v), d(u, v') + W(u, u')) \end{aligned}$$

，因此 Dijkstra 的每轮迭代中只需要更新 $O(V)$ 个标号，算法复杂度降至 $O(V^3)$。

3 2007 Tokyo

3.1 A. Consanguine Calculations

3.1.1 题意

多组询问：给定父母的血型，判断孩子可能的血型；或者给定父母一方和孩子的血型，判断父母另一方的血型。

3.1.2 题解

枚举未给定的人的血型，$O(1)$ 判断。

3.2 B. Containers

3.2.1 题意

有 $S = 26$ 条船和 N 件货物，每件货物属于一条船，货物按照给定的顺序到达码头。每当一件货物到达码头，你可以将它自己放成一堆，或者将它放到之前的某一堆的顶端。在所有货物到达码头之后，船按给定的顺序到达码头。每条船到码头之后，会将属于自己的货物调走。此时属于它的货物必须处于所在每一堆的顶端。

找到一种符合题意的分配方式，使得所有货物组成的总堆数最少。

$N \leq 4000$。

3.2.2 题解

将货物按照到达时间排序之后，可以注意到以所属船的编号为序，每堆货物构成一个不上升子序列。问题转化为使用最少的不上升子序列覆盖整个序列。

根据 Dilworth 定理，答案为原序列的最长上升子序列长度。由于货物所属船的编号范围很小，我们可以在 $O(SN)$ 时间内计算答案。

3.3 C. Grand Prix

3.3.1 题意

给定平面上的 N 个点 P_1 .. P_n。将整个图形顺时针 / 逆时针旋转最小的角度，使得将图形画在与水平面相交呈 θ 的平面上，使得平面的 y 轴与交线重合之后，图形满足 $z_i \leq z_{i+1}$。

$N < 10^4$。

3.3.2 题解

如果 $\theta = 0$，任意图形都满足条件。

考虑 $\theta \neq 0$ 的情况。我们需要对图形进行旋转，使得 $x_i \leq x_{i+1}$。首先考虑逆时针旋转。设转角为 α，于是 $x_i \cos\alpha + y_i \sin\alpha < x_{i+1} \cos\alpha + y_{i+1} \sin\alpha$，由此可得 $\cot\alpha$ 的取值范围，显然我们需要取其最大值。顺时针旋转等价于将图形关于 x 轴翻转之后逆时针旋转。

时间复杂度 $O(N)$。

3.4 D. Jacquard Circuits

3.4.1 题意

给定一个顶点为整点的多边形 P。设 P_1 为与 P 相似且顶点为整点的最小多边形，P_2 为与 P 相似的第二小整点多边形，..，P_k 为与 P 相似的第 k 小整点多边形，求出 P_1，$P_2..P_M$ 内的整点数之和。

$M < 10^6$，顶点数 $N < 1000$。

3.4.2 题解

首先我们需要求出 P_1。可以注意到 P 是由 P_1 缩放 $gcd(x_i - x_{i+1}, y_i - y_{i+1})$ 倍得到的。

根据匹克定理，一个整点多边形内的整点数为 $S - \frac{I}{2} + 1$，其中 S 表示其面积，I 表示多边形边上的整点数。分别考虑对于 $P_1..P_m$，这三部分的和：第一部分为 $\sum i^2 S_1$，第二部分为 $\sum iI_1$，第三部分为 M。因此，求出 P_1 的面积与整点数之后我们可以 $O(1)$ 得到答案。P_1 的面积可以用叉积在 $O(N)$ 时间内求出，边上的整点数等于相邻顶点横纵坐标之差的最大公约数，可以在 $O(N \log W)$ 时间内求出。

3.5 E. Collecting Luggage

3.5.1 题意

机场的行李传送带可以用简单多边形描述，其中行李入口在一个顶点上。行李匀速沿多边形边缘转动。一位乘客站在多边形外，当他的行李从入口进入时，他立即沿折线段匀速走到某个位置，使得他到达该点时行李恰好也到达该点。求他所用的最短时间。

多边形点数 $N \leq 100$；保证乘客速度快于传送带。

3.5.2 题解

由于乘客的速度快于传送带，假如他能在 T 时刻与行李会和，在 $T' > T$ 时刻他也必然可以与行李回合。所以会和时间可以二分得到。

对于给定的会和时间，我们可以计算行李在该时间到达的位置，计算乘客到该位置的最短路。可以注意到折线的端点必然是多边形的顶点，所以我们只需要在 $N + 2$ 个点的图中计算最短路。

3.6 F. Marble Game

3.6.1 题意

在 $N \times N$ 的棋盘的格子中有 M 个玻璃球和 M 个洞，每个玻璃球有唯一的一个洞与之对应。一部分格子之间的边线上有障碍物。每次操作可以将棋盘的一边竖起，让玻璃球滚向那一边。玻璃球不能越过障碍物或其它球；遇到有洞的格子时玻璃球会掉进去，之后这个洞会消失。求最少的操作次数，使得每个玻璃球都会掉进对应的洞中。

$N \leq 4$。

3.6.2 题解

注意到状态非常稀疏，可以直接 BFS。

3.7 G. Network

3.7.1 题意

有一些消息需要接收，每条消息由多个数据包组成。所有的数据包按照乱序传入，你可以利用一个缓冲区将它们按正确顺序传出（消息按任意顺序传出，但组成每条消息的数据包必须按顺序连续传出）。每收到一个数据包时，可以将它放入缓冲区，或传到输入端；可以在任意时刻将缓冲区中的任一消息传到输入端。给定数据包的传入顺序，找到一种消息的传出顺序使得所需缓冲区的大小最小。

消息数 $N < 6$；数据包数 $M \leq 1000$。

3.7.2 题解

枚举消息的传出顺序，模拟计算对于特定的传出顺序，所需的缓冲区大小。复杂度 $O(N!M)$。

3.8 I. Problem: Water Tanks

3.8.1 题意

N 个圆柱形水箱排成一排，每个水箱的底面积均为 $1m^2$，相邻的两个水箱之间用水平细管连接。除第 1 个之外，水箱与大气隔绝。给定所有水箱与细管的高度，求从第一个水箱至多能灌入的水量。

$N \leq 10$；设连接 $i-1$ 和 i 的细管高度为 C_i，则有 $C_i < C_{i+1}$。

3.8.2 题解

枚举最后一个有水的水箱 i，于是可以确定所有水箱中空气的分布情况：用压强与体积的积表示空气量，则初始时（水箱 2 中水的高度与 1 中高度相同时）水箱 2..N 中空气的总量 $\sum_{1<i\leq N} H_i - C_i$[3]；如果水箱 3 中的水位高度超过了 C_2，那时空气在水箱 2 与 3..N 间按体积比分配；以此类推。

确定每个水箱中空气的量之后，对于前 $i-1$ 个水箱，由于它们中的水是联通的，我们可以用水位表示水面处的压强，根据理想气体状态方程解出水位。对于第 i 个水箱，我们还需要判断其中的水与 $i-1$ 不联通的情况。

时间复杂度 $O(N)$。

3.9 J. Tunnels

3.9.1 题意

给定无向图 $< V, E >$，A 从点 s 出发，试图到达点 t。B 想要阻止 A 到达 t。A 与 B 轮流行动，B 先操作；B 在每一轮中可以从图中删除任意多的边，A 在每一轮中可以前往他所在点的任意相邻点。AB 二人均了解对方的操作。

A 的目标是最大化 B 删去的边数，B 的目标是最小化这一数量。求二人都使用最优策略时，B 需要删除的边数。

$N \leq 50$，100 组数据捆绑测试。

[3]忽略计量单位等

3.9.2 题解

经过简单的尝试，我们可以想到这样的做法：

设 C_u 为 u 到 0 的最小割，D_u 为间谍从 u 出发时，需要删掉的最少边数。使用类似 Dijkstra 的标号设定法迭代。初始时 $D_u = C_u$。考虑 D 值未确定且 D_u 最小的 u，其当前 D 值与所求最小值已经相等，或者说其 D 值已经确定；对于其它的点 v，我们用 “将 v 与 0 和所有 D 值已经确定的点隔开的最小代价 + 已经确定的 D 值中的最大值” 更新其 D 值，并重复这一循环。

这一算法显然可以用至多 N^2 次最小割计算实现。接下来我们需要证明，这样计算出的 D 值是正确的。

首先，我们证明它是实际值的上界。

引理 3.1. *按照被确定的顺序，D_u 是单调不减的（显然）。*

定理 3.1. *存在一种方案，使得间谍从 u 出发时，需要删去的边数的最大值为 D_u。*

证明. 以 D 值为序使用归纳法。

对于 D_u 最小的点 u，$D_u = C_u$，所以我们只要删掉 u 与 0 的最小割集即可。

假设对某个 d，$D_u \le d$ 的点的方案已经得到。设 S 为这样的点组成的集合。我们考虑间谍从 $D_u = d + 1$ 的点 u 出发的方案。在间谍出发之前，我们删去一部分边，使得所有连接 u 和 0 的路径必须经过 S 中的点。于是，间谍必然要经过一个属于 S 的点 v，而对点 v 我们已经构造出了代价为 D_v 的方案。当他第一次到达这样的 v 时，我们使用这一方案。

显然，这样的方案需要删去不超过 D_u 条边。 □

接下来我们证明计算出的 D 值是实际值的下界。

引理 3.2. *考虑在图 G 中删去一条边 (s_i, e_i) 之后得到的图 G'。设存在 u 使得 $D'_u < D_u - 1$，则存在 (s_i, e_i) 的端点 p，使得 $D_p < D_u, D'_p < D'_u$。*

证明. 以下设 $S'(t) = \{x | D'_x < D'_t\}, S(t) = \{x | D_x < D_t\}$。

考虑此前确定的 D 值最小的点 v，满足 $D'_v = D_v - 1$。很明显，这样的 v 必然存在。于是有 $S(v) = S'(v)$，$\forall x \in S(v), D_x = D'_x$。

考虑图 G 中，间谍在 v 时的方案：删去 x 条边，使得连接 v 和 0 的路径必须经过 $S'(v)$ 中的点。设 $y = \max_{x \in S'(v)}(D'_x)$，于是 $D'_v = x + y$。在 G 中删去相同的 x 条边之后，有两种可能的情况：

连接 v 和 0 的路径必须经过 $S'(v)$ 中的点，亦即，必须经过 $S(v)$ 中的点。计算 D_v 时应有 $D_v \le x + y = D'_v$，矛盾；

存在一条连接 v 和 0 的路径不经过 $S'(v)$ 中的点，于是这条路径必然经过 (s_i, e_i)。因此，另外删去 (s_i, e_i) 后，该边的一个端点 p 与 v 联通。反证可得此时 p 必须经过 $S(v)$ 中的点才能到达 0，亦即 $D_p \le x + 1 + y = D_v < D_u$，$D'_p < D'_u$。

□

引理 3.3. *考虑在图 G 中删去一条边 (s_i, e_i) 之后得到的图 G'。不存在 u 使得 $D'_u < D_u - 1$。*

证明. 设此结论不成立，u 为使得 $D'_u < D_u - 1$ 且 D_u 最小的点。于是有 $S'(u) \subset S(u)$，$\max_{t \in S(u)}(D_t) \le \max_{t \in S'(u)}(D'_t) + 1$。根据引理3.2，存在 (s_i, e_i) 的端点 p 使得 $p \in S(u)$。

考虑在 G' 中，间谍在 u 时的方案：删去 x' 条边，使得连接 u 和 0 的路径必须经过 $S'(u)$ 中的点。设 $y' = \max_{x \in S'(u)}(D'_x)$，于是 $D'_u = x' + y'$。在 G 中删去相同的 x' 条边之后，有两种可能的情况：

连接 u 和 0 的路径必须经过 $S'(u)$ 中的点，亦即，必须经过 $S(u)$ 中的点。那么，$D_u \leq x + \max_{t \in S(u)}(D_t) \leq x + y + 1 = D'_u + 1$，矛盾；

存在一条连接 u 和 0 的路径不经过 $S'(u)$ 中的点，于是此路径必然经过 (s_i, e_i)，由 $p \in S(u)$，该路径经过了一个 $S(u)$ 中的点。同上得到矛盾。 □

引理 3.4. *对于任意图 G 中的点 u，存在连接 u 和 0 的路径使得路径上 D 值的最小值不小于 D_u（显然）。*

定理 3.2. *不存在一种方案，使得间谍从 u 出发时，只需要删除 $D_u - 1$ 条边。*

证明. 以 D 值为序使用归纳法。

$D_u = 1$ 时，由引理3.4可知，存在一条从 u 到 0 的路径。因此不存在不需要删边的方案。

设结论对于 $D_u < d_0$ 的所有 u 成立，考虑 $D_u = d_0$ 的某个 u。由引理3.4可知，存在一条从 u 到 0 的路径满足其 D 值最小值不小于 d_0。令间谍沿该路径行进，直至第一次删边。设间谍在点 v 时，有 k 条边被删去。由引理3.3可知，$D'_v \geq D_u - k$；由归纳假设可知，在新图中不存在一种方案，使得从 v 出发时只需删除 $D_u - k - 1 \leq D'_v - 1$ 条边。 □

综上可得，计算出的 D 值与实际值相同。

定理 3.3. *对于任意点 u，D_u 为间谍从 u 出发，需要删去的边数的最小值。*

证明. 由定理3.1与定理3.2立即可得。 □

4 2008 Banff

4.1 A. Air Conditioning Machinery

4.1.1 题意

在 $N \times M \times K$ 的立方体中，用最少的 L 型管联通表面的两个方格，满足所有的 L 形管不相交。

L 形管指一段长度为 3 的直管与一段长度为 2 的直管尾端垂直相接得到的管道。L 形管可以以任何方式放置，只要满足其占据的每一个格子都在立方体内。

如果需要超过 6 个管子才能连接，输出“Impossible”。

4.1.2 题解

每个管子只有 8 种可能的形态，所以直接枚举。

4.2 B. Always an integer

4.2.1 题意

给定一个整系数多项式 $F(x)$ 和整数 d，判断是否对所有的 $x \in N$，都有 $F(x)/d$ 为整数。

$F(x)$ 的最高次项指数 N 不超过 100。

4.2.2 题解

设 $G(x) = F(x)/d$，$\Delta F(x) = F(x) - F(x-1)$，$\Delta^{(n)} F = \Delta(\Delta^{(n-1)} F)$。于是（对所有的整数 x，下同）$G(x)$ 为整数当且仅当 $G(0)$ 为整数且 ΔG 恒为整数；而 ΔG 恒为整数当且仅当 $\Delta G(0)$ 为整数且 $\Delta^2 G$ 恒为整数，等等。所以我们只需要检查 G 的各阶差分在 $x = 0$ 时是否为整数。等价地，我们只需要检查 G 的任意连续 $N + 1$ 个值是否均为整数。

时间复杂度 $O(N^2)$。

4.3 E. Huffman Codes

4.3.1 题意

考虑对 N 个字母计算其哈夫曼编码[4]的过程。给定每个字母最终的编码，确定满足条件的字频方案数。

注意经字频在排序后相同的字频方案视作相同，例如 $\{(0, 49), (1, 51)\}$ 与 $\{(0, 51), (1, 49)\}$ 相同。

每个字母的字频为 1 到 100 之间的整数，且 $N \leq 20$。

4.3.2 题解

本题可以用暴力解决，自顶向下搜索哈夫曼树上每个节点的频度。

根据哈夫曼树的性质，可以想到以下优化：

- 第 i 层的节点的频度低于第 $i-1$ 层任意节点的频度，否则在构造过程中它会与 $i-1$ 层的节点合并。

[4] 维基百科

- 总是搜索当前树中频度最高的叶子的子节点，这样可以保证当前方案总是合法的。

算法的复杂度与解数同阶。由于解数不多，这样足以通过本题的测试数据。

4.4 F. Glenbow Museum

4.4.1 题意

一个所有内角均为直角的多边形可以这样用 01 序列来描述：从它的任意一个顶点出发，沿逆时针方向依次记录所有内角的大小，对每个角，如果其大小为 90 度则用 0 表示，否则用 1 表示。注意同一个多边形可能对应不同的 01 序列，同一个 01 序列也会对应不同的多边形（边的长度可以任取）。

给定 N，计算长度为 N，且存在一个能对应到它的星型域[5]的 01 序列数量。

4.4.2 题解

原条件等价于序列中 0 的个数恰比 1 的个数多 4，且不存在连续的 1（且首尾不能同时为 1）。

由此原问题可以递推解决。用 $dp[i][j][s][e]$ 表示首项为 s，末项为 e，长度为 i 且有 j 个 1 的合法序列数，枚举下一项转移。时间复杂度 $O(N^2)$。

4.5 G. Net Loss

4.5.1 题意

给定次数为 N 的多项式 $P(x)$，以及常数 c，求一个分段线性函数

$$f(x) = \begin{cases} a_1x + a_0 & x < c \\ b_1x + b_0 & otherwise \end{cases}$$

使得 $a_1c + a_0 = b_1c + b_0$，且 $\int_{-1}^{+1}(P(x) - f(x))^2$ 最小。

$N \leq 10$。

4.5.2 题解

设 $d = f(c)$，对积分式化简可以得到仅关于 a_1，b_1 和 d 的函数。枚举 d 之后，该函数可以分为独立的两部分，每一部分是关于 a_1 或 b_1 的二次函数；$O(1)$ 求得此时的最小值。也可以直接推导极值的公式。

4.6 H. Painter

4.6.1 题意

给定 N 个三角形，完成以下任务：

- 判断是否存在两个三角形相交（共点）。

[5]维基百科

- 如果没有三角形相交，计算用以下方式构造的森林中节点的最大深度：对每个三角形，构造一个点；如果三角形 i 是包含三角形 j 的最小三角形，在森林中令 j 的父节点为 i。

$N \leq 10^5$。

4.6.2 题解

在没有三角形相交的情况下，森林的构造可以用扫描线算法解决。维护与当前扫描线相交的所有三角形。插入三角形 i 时，找到当前在它下方的三角形中，在扫描线的投影最靠上的一个。设该三角形为 j，易证 i 的父节点一定是 j 的某个祖先。所以在 j 的祖先列表中二分找到最后一个包含 i 的三角形即为所求。

判断三角形相交可以在上述扫描中顺便完成：只需要在插入一个三角形时，检查每条边是否与（在扫描线上的投影）离它最近的线段相交。

算法复杂度 $O(N \log N)$。

4.7 I. Password Suspects

4.7.1 题意

给定 N 个长度不超过 10 的字符串，求由仅由小写字母组成、长度为 L 且包括全部给定字符串的字符串数。如果这样的字符串数量不超过 42，输出所有这样的字符串。

$N \leq 10, L \leq 100$。

4.7.2 题解

由于 N 很小，我们可以在 AC 自动机上进行状态压缩动态规划。设 $f[l][v][S]$ 表示长度为 l，（最长）后缀在自动机上对应的节点为 v，包含给定串的子集 S 的字符串数量，枚举下一个字符进行转移。复杂度 $O(|\Sigma| \sum len_i L 2^N)$。

4.8 J. The Sky is the Limit

4.8.1 题意

N 座等腰三角形的山排在一条直线上，求形成的轮廓线的长度。

轮廓线指图形在直线的每个位置的最高点连成的折线段。

$N \leq 100$。

4.8.2 题解

显然折线段的拐点只能出现在三角形之间的交点，或三角形的端点上。这样的点的数量为 $O(N^2)$。计算所有这样的点，对每个点计算图形在该位置的最高点。

时间复杂度 $O(N^3)$。

4.9 K. Steam Roller

4.9.1 题意

给定一个网格图，相邻的格点之间有带权边。给定起点终点，求连接这两点的最小权路径。

一条路径的权值用以下方法计算：对路径上的每条边，如果它与前后某条边的方向不同，或者如果它是路径中的第一条或最后一条边，则累加它的权值的两倍；否则累加它的权值。

网格的行列数 N 不超过 100。

4.9.2 题解

将每个格点拆成 4 个点，分别表示从该点出发，下一条边方向为上 / 下 / 左 / 右时，到终点的最短路。在新图上计算最短路。

算法复杂度 $O(N^2 \log N)$。

5 2009 Stockholm

5.1 A. A Careful Approach

5.1.1 题意

给定 N 个区间，在每个区间中选择一个数，使得选择的所有数之间最小的差值最大。

$N \leq 8$。

5.1.2 题解

如果每个区间中选的数的顺序已经确定，我们可以二分差值，然后贪心验证，让当前区间中选择的数尽量小即可。

因为 N 很小，我们可以枚举所有可能的顺序，然后重复以上算法。

5.2 B. My Bad

5.2.1 题意

给定由输入、输出、四种门电路（与、或、非、异或）组件构成的逻辑电路，以及若干种输入对应的电路输出。已知电路中有不超过一个元件有故障，故障可能是总是输出一种信号，或总是输出与正确输出相反的值。找出这个故障元件。

组件数 $N \leq 46$，输入输出组数 $M \leq 36$。

5.2.2 题解

枚举出错的元件以及出错方式，动态规划验证测试的输入输出是否相符。复杂度 $O(N^2M)$。

5.3 D. Conduit Packing

5.3.1 题意

给定四个圆，求一个半径最小的大圆使其可以包含给定的四个圆，且所有的五个圆互不相交。

5.3.2 题解

枚举四个圆的位置关系（极角顺序），二分大圆的半径 R 进行验证。

一种验证的思路是，固定四个圆中最大的一个的位置，计算其它三圆到圆心的最小极角。由几何知识可知，如按极角序圆 i 在 j 之后，则

$$\theta_i \geq \theta_j + \arccos(\frac{(R-r_i)^2 + (R-r_j)^2 - (r_i+r_j)^2}{2(R-r_i)(R-r_j)})$$

，且对每个 i，由之前所有 j 确定的 θ_i 下界中必有一个是确界。

最后，我们需要验证所有圆的极角是否与圆 1 相容。我们可以增加新圆 $R_5 = R_1$，验证 θ_5 是否超过 2π。

5.4 E. Fare and Balanced

5.4.1 题意

给定一个边带权的有向无环图，增加一些边的权值，使得

- 从 1 到 N 的所有路径权值相等；
- 从 1 到 N 的任意一条路径上，至多有一条权值被增加过的边；
- 从 1 到 N 的距离最小。

$N, M < 10^5$。

5.4.2 题解

设 $f(s,t)$ 表示原图中从 s 到 t 的所有路径权值是否相等，$d(s,t)$ 表示原图中从 s 到 t 的最长路。

定理 5.1. *如果存在 v 使得 $f(1,v) = f(v,n) = 1$，则不存在合法方案。*

证明. 假设存在一种方案，则在方案中我们显然需要修改 1 到 v 的某条路径上的边，以及 v 到 n 的路径上的边。连接这两条路径，可以得到一条从 1 到 n，有至少两条权值增加过的边的路径。 □

上述条件可以在 $O(N+M)$ 时间内完成检验，只需要计算从 1 和 n 出发，到每个点的最长路与最短路即可。

定理 5.2. *对于不满足上述条件的图，考虑所有的满足以下条件的边 (u,v,w)（以下称之为 "关键边"）：*

$$\begin{aligned} f(1,u) &= 0 \\ f(1,v) &= 1 \\ d(1,u) + w + d(v,n) &< d(1,n) \end{aligned}$$

将所有关键边的边的权值增加 $d(1,n) - (d(1,u) + w + d(v,n))$，必然是一种合法且最优的方案。

证明. 考虑方案的合法性。对于任意一条边 (s,t)，显然 $f(1,s)$ 蕴含 $f(1,t)$。因此，在任意一条路径上至多包含一条关键边，也就是至多有一条权值被增加过的边；又该方案中从 1 到 n 的所有路径的权值均等于 $d(1,n)$，因此该方案为合法且最优的。 □

利用之前计算的最长路与最短路信息，上述方案可以在 $O(N+M)$ 时间内构造。至此，问题得到解决。

5.5 F. Deer-Proof Fence

5.5.1 题意

给定平面上的 N 个点，确定若干段闭合曲线，使得曲线外的任意点到给定点集的最小距离不小于 M，且曲线长度之和最小。

$N < 10$。

5.5.2 题解

考虑需要用一段闭合曲线包含所有点的情况。问题等价于给定 N 个半径为 M 的圆，找出包围它们的最短闭合曲线。我们应当计算点集的凸包，并将凸包中每条边 “外推”M 的距离，用半径为 M 的圆弧连接相邻点。如果需要一个严格的证明，可以考虑圆是正 K 边形（$K \to \infty$）的情况。

回到本题。设 $f[S]$ 为用一段曲线包围给定点集的子集 S 的最小长度，$g[S]$ 为用若干段曲线包围 S 的最小长度。于是

$$g[S] = \min(f[S], \min_{S' \subset S}(f[S'] + f[S - S']))$$

。算法复杂度为 $O(3^n)$。

5.6 G. House of Cards

5.6.1 题意

一种纸牌游戏的规则如下所述：

1. 游戏在半副标准纸牌（两种颜色，每种颜色包含 13 个等级）的子集上进行。
2. 游戏开始时，玩家将选择的纸牌洗牌，选出最上面的 8 张，从左到右将它们摆成 4 个 “山峰”。剩下的排正面向上放成一排备用。
3. 每个玩家被分配一种颜色。用于摆成山峰的第一张牌的颜色对应的玩家先走。
4. 玩家交替操作。每次操作的玩家取出备用牌最前面的一张，并选择以下一种方式：
   - 持有这张纸牌直到下次操作（以下称该牌为 “被持有的纸牌”）。
     - 任何时刻玩家持有的纸牌数不能超过 1。
   - 用刚抽取的纸牌或被持有的纸牌覆盖在两个山峰之间的山谷，形成一个 “基底”。如果还剩下一张牌，那么这张牌就被持有。
   - 把 2 张纸牌放在基底上面，形成一个山峰（其中一张纸牌一定是一张被 “持有” 的纸牌）。
5. 备用牌抽完时游戏结束。
6. 如果玩家新添加的纸牌与之前的牌堆形成了一个三角形，那么三角形中出现较多的一种颜色对应的玩家的分数会增加三角形中纸牌的等级之和；游戏结束时，如果玩家手中有牌，则如果牌的颜色与玩家对应颜色相同，该玩家的分数增加牌的等级；否则，该玩家分数减少牌的等级数。游戏目标是最大化自己与对方的分数差。

给定选择的纸牌集合以及某种颜色对应的玩家，计算它与对方最大的分数差，假设双方均使用最优策略。

5.6.2 题解

直接搜索双方的策略。添加 alpha-beta 剪枝即可通过所有可能的测试数据。

alpha-beta 剪枝 在最大化双方分差的游戏的搜索中常用的剪枝。其策略可以简单描述如下：考虑当前局面 S 的两个后续局面 S_1、S_2。设在局面 S 中操作的玩家为 A，另一位玩家为 B，玩家 A 需要最大化分差[6]，B 需要最小化分差。如果在 S_1 中，B 使用最优策略可以得到 K 的分差，而在 S_2 中，B 使用某一策略可以得到 $K' < K$ 的分差，则为了最大化分差，在局面 S 中玩家 A 肯定不会使游戏进入后续局面 S_2，因此，继续搜索 S_2 的其它后续是无意义的，可以直接退出 S_2 的搜索。

5.7 H. The Ministers' Major Mess

5.7.1 题意

有 N 个大臣和 M 个议案，每位大臣对不超过四个议案投票表示赞成或反对。

判断是否存在一种议案的通过方案，使得每位大臣都有超过一半的建议得到满足。如果这样的方案存在，找出所有必须通过或必须否决的议案。

$N < 500, M < 100$。

5.7.2 题解

由于每位大臣的投票数不超过 4，原条件等价于 "每位大臣至多有一项建议被否决"。亦即，如果大臣 i 的投票 T_{ij} 被否决，其所有其它投票都必须通过。于是问题转化为 2-SAT：设 f_i 表示议案 i 是否被通过，于是

$$f(T_{ij}) \vee f(T_{ik}), j \neq k$$

必须得到满足。

2-SAT 解的存在性很容易判断。如果 $f_i \Rightarrow \neg f_i$，则议案 i 必须被否决；$\neg f_i \Rightarrow f_i$ 则议案 i 必须通过。由此，必须通过或否决的议案也可以判断。

取决于实现，算法复杂度为 $O(M^3)$ 或 $O(M^2)$。

5.8 I. Structs and Springs

5.8.1 题意

N 个矩形窗框放置在平面上。同一窗框的相对边之间用变长弹簧或定长轻杆相连；每一个窗框与包含它的窗框之间用四个弹簧或轻杆相连，分别在平行的四条边处固定。任意时刻，连接在统一窗框平行边上的弹簧的长度改变量成比例。没有窗框相交，且存在一个窗框包含其它所有的窗框。

M 次操作，每次改变最外层窗框的大小，计算此后其它窗框的大小，以及相对最外层窗框的位置。

$N \leq 500, M \leq 500$。

5.8.2 题解

构造窗框之间的包含关系，按照题意模拟即可。

[6]本段所有的分差均指 $A - B$ 的分差。

5.9 J. Subway Timing

5.9.1 题意

给定一棵边带权的树，你需要将每条边的权值 w 改为 $w \bmod 60$ 或 $-(60 - w \bmod 60)$。选择一种修改方案，使得修改后树上权值和的绝对值最大的路径的绝对值最小。

$N < 100$。

5.9.2 题解

定理 5.3. *答案不会超过 240。*

证明. 只需要证明，存在一种方案，使得对于任意 x 是 y 祖先的 (x, y)，x 到 y 的路径权值和 $W(x, y)$ 的绝对值不超过 120。
任意定根，从根 r 出发自顶向下确定所有边的权值。对于一条权值为 w 的边 (u, v) $(dep_u < dep_v)$，将如果 $W(r, u) < 0$ 则令 $w = w \bmod 60$，否则令 $w = -(60 - w \bmod 60)$。于是，对于任意 x $|W(r, x)| < 60$，对于任意 x 是 y 祖先的 (x, y) $|W(x, y)| < 120$。 □

根据这一性质，我们二分答案 W，用动态规划验证。设 $dp(x, T)$ 为以 x 为根的子树中，任意经过 x 的路径的权值和的绝对值不超过 W，且从 x 出发的最小和路径权值为 T 时，从 x 出发的最大和路径的最小权值，于是算法复杂度为 $O(NT^2 \log W)$。

5.10 K. Suffix-Replacement Grammars

5.10.1 题意

一个后缀替换文法由起始字符串 S 和若干规则组成。每个规则以 $X \to Y$ $(len(X) = len(Y))$ 的形式给出，表示在任何时刻，如果 X 是当前字符串的后缀，我们可以把它替换为 Y。

给定文法的起始字符串 S 和替换规则，判断该文法是否可以生成串 T；如果可以，确定最少的步数（最少的使用规则次数）。

$len(S) = len(T) \le 20$，规则数 $n \le 100$。

5.10.2 题解

问题相当于求一个点数极多的图中的最短路，其中每个结点对应一个可能生成的字符串。

可以发现该图可以被划分为若干同构的部分，其每一部分又可以被继续划分。例如以 'A' 开头长度为 3 的字符串对应的子图，和以 'B'、'C'、'D'…开头，长度为 3 的字符串对应的子图是同构的。所以，我们可以分别考虑每个长度的字符串对应的子图，求出其中的最短路，利用得到的信息计算更长的字符串对应子图的最短路。

注意到在长度确定的字符串对应的子图中，只有那些是某一规则或 S/T 的后缀的字符串是有意义的，所以每个子图的节点数不超过 $2n + 2$。使用 Floyd 算法计算最短路，算法的复杂度为 $O(len(S)n^3)$。

6 2010 Harbin

6.1 B. Barcodes

6.1.1 题意

给定一个条形码，求其对应的原串。

在条形码中，原串的每个字符会被编码为 5 个相邻的黑色区域，区域有宽区域与窄区域之分，前者的宽度是后者的两倍。字符与编码是一一对应的。

条形码最前面和最后面的 5 个区域分别对应开始标记和结束标记。它们的编码与所有字符均不同；开始标记和结束标记的编码均为“窄窄宽宽窄”。

在将原串编码前，我们会在原串最后添加两个校验字符。第一个校验字符的值由原串的字符运算得到，第二个字符的值由原串和第一个校验字符运算得到。

给定条形码中每个区域的宽度，判断其是否合法。如果合法，输出对应的原串。

区域数不超过 150。*宽度与真实值存在不超过 5% 的误差。*

6.1.2 题解

如果能够确定每个区域是宽区域还是窄区域，接下来只需要按题意检查每个编码即可。

考虑宽窄区域的确定。将所有区域按照宽度升序排序得到数组 $A_1..A_n$，于是 A_1 是窄区域，A_n 是窄区域。设 A_i 是最宽的窄区域，则

$$\frac{\max(A_n, 2A_i)}{1+5\%} \le \frac{\min(2A_1, A_{i+1})}{1-5\%}$$

。枚举 i 验证即可。

6.2 C. Tracking Bio-bots

6.2.1 题意

给定一个 $N \times M$ 个方格的棋盘，棋盘上有 K 排左右延伸的墙。一个机器人从右上角出发，每一步只能向左或向下走一格，且不能经过有墙的格子。统计不可能被机器人经过的格子数。

$N, M \le 10^6, K \le 10^3$。

6.2.2 题解

将棋盘离散化后 BFS 是 $O(K^2)$ 的。对于更大的数据范围，我们有 $O(K \log N)$ 的做法：

从上到下考虑每一行有多少格子不可能被经过。最上一行的情况是显然的。假设上数第 i 行中，不可能被经过的格子的列标号已被算出，为 $[l_1, r_1], [l_2, r_2]..[l_{k_i}, r_{k_i}]$，下面要计算第 $i+1$ 行的情况。

考虑一个连续不包含墙的子段 $[l, r]$（省略行标号，下同），如果格子 $x \in [l, r]$ 可以被到达，$[l, x)$ 也必然可以。所以子段中的可达区域可以写成 $[l, x]$ 的形式，且显然 $x = \min(r, \min(i : i \ge l, \exists j\ i \in [l_j, r_j]))$。升序考虑每个这样的子段，上式中 x 和 j 的值不减，所以可以在一趟 $O(k_i)$ 的循环中计算所有的 x 值。

算法复杂度为 $O(\sum_{1 \le i \le N} \max(1, k_i) + K \log K) = O(N + K \log K)$。注意连续的空行可以压缩成一行，由此复杂度降至 $O(K \log K)$。

6.3 D. Castles

6.3.1 题意

N 个城堡连成树型，你需要将它们全部占领。占领城堡 i 需要至少 A_i 名士兵参与进攻，有 M_i 名士兵会阵亡，K_i 名 i 名士士兵之后必须驻守在 i。

计算从任意城堡开始占领所有城堡需要的最少士兵数。

$N \leq 100$。

6.3.2 题解

枚举最早占领的城堡，动态规划计算答案。

设 F_i 为（从 i 出发）占领以 i 为根的子树所需的士兵，G_i 为占领过程中驻守与阵亡的士兵数之和。假设 i 的所有孩子 $C_1..C_k$ 的 F, G 值已被计算，现在要计算 F_i 和 G_i。

显然 $G_i = \sum(G_{C_j})$，考虑 F_i。假如子树的占领顺序已经确定，我们先占领 C'_1，之后是 C'_2, C'_3 .. C'_n，则此时需要的士兵是恒量的，为 $\max_i(\sum_{j<i} G_{C'_j} + F_{C'_i})$，于是，

$$F_i = \max(A_i, M_i + K_i + \min_{\text{C' 是 C 的排列}}(\max_i(\sum_{j<i} G_{C'_j} + F_{C'_i})) \qquad (*)$$

，因为我们还要占领城堡 i。

尝试从 C 的任意一个排列开始，调整得到最优的 C'。如果交换 C'_i, C'_j 有可能得到更优解，那么他们对答案的贡献的最大值应该不增，亦即，

$$\max(\sum_{k<i} G_{C'_k} + F_{C'_j}, \sum_{k<j} G_{C'_k} + F_{C'_i}) \leq \max(\sum_{k<i} G_{C'_k} + F_{C'_i}, \sum_{k<j} G_{C'_k} + F_{C'_j})$$

。[7] 设 $a = C'_i, b = C'_j, X = \sum_{k<i} G_{C'_k}, Y = \sum_{i<k<j} G_{C'_k}$，则

$$\max(X + F_b, X + G_b + Y + F_a) \leq \max(X + F_a, X + G_a + Y + F_b)$$

亦即

$$\max(F_b, G_b + Y + F_a) \leq \max(F_a, G_a + Y + F_b)$$

。

考虑不等式不成立的条件。很明显不可能是 $F_b > \max(F_a, G_a + Y + F_b)$，所以 $G_b + Y + F_a > \max(F_a, G_a + Y + F_b)$，所以 $G_a - F_a > G_b - F_b$。

所以不等式成立等价于 $G_a - F_a \leq G_b - F_b$。也就是说，最优的 C' 是将 C 按照 $F_{C_i} - G_{C_i}$ 排升序得到的。于是将 C 排序之后，按 $(*)$ 式就可以得到 F_i 了。

于是对固定的根，动态规划复杂度为 $O(N \log D)$；算法最终的时间复杂度为 $O(N^2 \log D)$，其中 D 是树中的最大度数。

6.4 F. Contour Mapping

6.4.1 题意

给定一个全等正三角形组成的网格，网格中每个顶点的高度已知。假设每个三角形区域都是平面，计算海拔为 kD ($k \in \mathrm{N}$) 的等高线长度之和。

三角形数 $N \leq 10^4$，$D \leq 10^3$。

[7] 注意左边的 C' 是交换 C'_i, C'_j 之后的，右边是交换前的。

6.4.2 题解

分别统计每个三角形内等高线的总长。如果某个三角形的边界是等高线，将重复计算的部分从答案中减去。

算法复杂度 $O(N)$。

6.5 G. The Islands

6.5.1 题意

平面上有 N 个点 $P_0..P_{N-1}$，P_0 在最左边，P_{N-1} 在最右边。一个人从 P_0 出发，先一直向右走到 P_{N-1}，再一直向左走回 P_0，途中经过 $P_1..P_{N-2}$，且在往返两趟中分别经过 P_a, P_b 两个点。

给定 $P_0..P_{N-1}$ 以及 a, b，求满足上述条件的最短路。

$N \leq 100$。

6.5.2 题解

问题等价于选择两条从 P_0 到 P_{N-1} 的路线，满足横坐标均不减，且 P_a, P_b 分别在两条路线中。

不失一般性，假设 P_a 在第一条中，P_b 在第二条中。设 $f(i,j)$ 表示两条路线的最后一个点分别是 i 和 j 的情况下的最短路，枚举 $\max(i,j)+1$ 属于哪条路径转移，$\min(f(i,n)+dis(P_i,P_n), f(n,j)+dis(P_j,P_n))$ 即为所求。

算法复杂度 $O(N^2)$。

6.6 I. Robots on Ice

6.6.1 题意

给定 $N \times M$ 的方格网络，求满足以下条件的从 $(0,0)$ 到 $(0,1)$ 的哈密顿路径数：路径恰在 $\lfloor \frac{imn}{4} \rfloor$ 步经过检查点 $P_i (1 \leq i \leq 3)$。

$N, M \leq 8$。

6.6.2 题解

本题可以用 DFS 解决。添加以下剪枝足以通过本题的测试：

- 如果不可能按时到达下一个检查点（用当前位置与检查点的曼哈顿距离比较），或是提前到达了下一个检查点，则当前路径不合法。
- 如果从当前位置沿某个方向扩展会将未访问的格子分成非空的两部分，则这一扩展不合法。

6.7 J. Sharing Chocolate

6.7.1 题意

有一块 $N \times M$ 格的巧克力要分给 K 个人，第 i 个人要求他的巧克力包含 K_i 块。每次分割可以沿一条行或列的分割线将当前的某块巧克力切开。判断是否存在一种分割方案。

$N, M \leq 100, K \leq 15, \sum K_i = NM$。

6.7.2 题解

很容易想到用 $f(n,m,S)$ 表示 $n\times m$ 的巧克力是否恰好可以满足集合 S 中人的要求。注意到 $f(n,m,S)=1$ 的必要条件是 $\sum_{i\in S}K_i=nm$，所以对于固定的 n,S，至多有一个 m 满足 $f(n,m,S)=1$（$m=\frac{\sum_{i\in S}K_i}{n}$，如果存在）。所以我们可以把状态表示改为 $f(n,S)$，含义是否存在 n 行的巧克力恰可以满足 S 的要求，于是 $f(N,\{1..K\})$ 即为答案。

枚举下一刀是沿行切还是沿列切，再枚举切下后，新形成的两块巧克力对应的人的集合。此时应该计算的状态也就确定了。

算法复杂度 $O(N3^K)$。

6.8 K. Paperweight

6.8.1 题意

给定一个六面体，其中某个位置包含一个识别芯片（视为点）。将六面体放到平面上，使其是稳定的，且识别芯片离放置平面的距离最小 / 最大。

稳定的定义是，在重心在任一方向移动不超过 0.2 的距离的情况下，六面体均不会移动。

6.8.2 题解

枚举六面体与平面接触的点构成的支撑面更新答案。

稳定的定义等价于重心在平面上的投影在支撑面之内，且到支撑面任一边的距离不小于 0.2。

注意支撑面不一定是六面体的面。正确的做法是枚举所有从 5 个点中选 3 个点的组合，如果剩下两个点在这三个点构成平面的同测，就计算以其为支撑面的答案。

7 2011 Orlando

7.1 A. To Add or to Multiply

7.1.1 题意

某种处理器仅包含一个寄存器和两种指令：将寄存器中的值加 a（'A'），或将寄存器中的值乘 m（'M'）。

给定两个区间 $[p,q]$ 和 $[r,s]$，寻找一个最短的指令序列，使得对于 $[p,q]$ 中的任意数 x，将 x 载入寄存器经该序列处理之后，寄存器中数值均在 $[r,s]$ 范围内。

$0 < p,q,r,s < 10^9, a > 0, m > 0$。所有数值均为整数。

7.1.2 题解

$m = 1$ 时的做法是显然的。

$m > 1$ 时，序列中不能包含超过 $\log S$ 个 'M'，所以我们枚举 'M' 的数量 m，计算此时 'A' 的最小数量。

设数 x 经过序列处理后变换到 $M^m x + aA$，如果 m 和 a 均已固定，我们可以从前往后扫描 M 的序列，贪心地在相邻的 'M' 之间插入尽量多的 'A'。例如 $M = 3, m = 3, a = 19$ 时我们得到了"M 2A M M A"。

现在只有 m 固定的，但是 a 的范围 $[l,r]$ 可以计算。可以发现 'A' 的数量关于 a 转换为 M 进制之后各位之和单增，所以之后 'A' 的数量应当是 $l, m\lceil l/m\rceil, m^2\lceil l/m^2\rceil ...$ 这一序列中的某一项。也就是说，对于固定的 m，a 只有 $\log l$ 个值，我们可以逐一尝试。

更新答案的时间与（压缩相同项之后的）序列长度同阶。由于压缩后序列长度为 $O(\log S)$，算法的复杂度为 $O(\log^3 S)$。

7.2 B. Affine Mess

7.2.1 题意

给定点 P, Q, R 和 P', Q', R'，找到一种旋转，缩放和平移操作的组合方案，使得 P, Q, R 经处理后与 P', Q', R' 重合。

坐标的平移量只能是整数；缩放系数为非零整数，且横坐标和纵坐标的系数可能不同；旋转操作是最先进行的，操作方法为在以原点为中心，边长 20 的正方形上选择一个整点，旋转坐标轴使得 x 轴通过该点，且原先单位长度不变。旋转之后所有坐标舍入到最近的整数。

如果有它解，判断他们是否相容。相容的定义是，任意点 P 经过两种方案变换之后均变换到同一点 P'。

7.2.2 题解

枚举初态和末态的对应关系以及旋转的方法，之后平移和缩放的方法可以解线性方程组得到。显然如果某一方程组有多解，原题必然有不相容的多组解。

另外，容易证明如果两个操作序列的旋转角不关于坐标轴或原点对称，则这两种操作必不相容。（考虑坐标足够大的某一个点。舍入和平移操作对它到原点的极角的影响是可以忽略的。所以

它在两种变换下会被变换到不同的点。）所以，对得到的所有操作序列，忽略舍入过程将它用矩阵表示，如果有两个操作序列对应着不同的矩阵，则存在不相容的解；否则所有的解都是相容的。

7.3 C. Ancient Messages

7.3.1 题意

给定六种象形符号和一个 $N \times M$ 的点阵图，判断其中每种符号出现的次数。

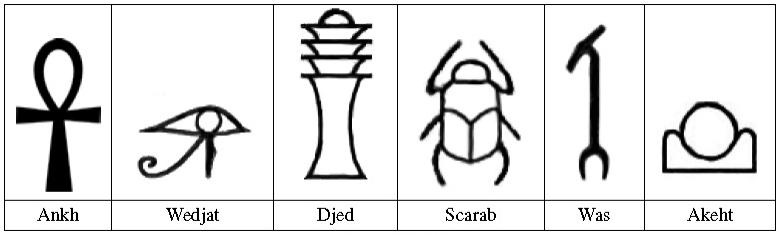

Figure 2: 象形符号图

$N, M \leq 2000$，两个符号之间的距离至少为 1，符号可能有拉伸和翻转。

7.3.2 题解

从图中可以发现，每种符号的黑色部分联通，且与黑色部分相邻的白色联通块数量两两不同。所以 floodfill 之后，统计每个黑色联通块相邻的白色联通块数量即可。复杂度 $O(NM)$。

7.4 D. Chips Challenge

7.4.1 题意

给定 $N \times M$ 的网格，其中一些格子上有障碍，一些格子上已经放置了棋子，剩余的格子是空的。

在空格上放置尽量多的棋子，满足

- 第 i 行和第 i 列的棋子数相等；
- 每行的棋子数不能超过总棋子数的 a/b。

$N, M < 40$。

7.4.2 题解

不考虑第二个约束，本题可以用上下界费用流解决。对每行每列各建一个点，如果第 i 行第 j 列的格子为空，连边 $(row[i], col[j], 0, 1, 1)$[8]；如果该格已被放置棋子，连边 $(row[i], col[j], 1, 1, 1)$；另外连边 $(col[i], row[i], 0, \infty, 0)$。这一网络上的最大费用可行流即为所求。算法的正确性可由流守恒性得到。

[8]以下在上下界费用网络中用边 (a, b, f_0, f_1, c) 表示从 a 到 b，容量下界为 f_0，上界为 f_1，费用为 c 的边；在无下界费用网络中，省略 f_0。

为了满足第二个约束，我们可以枚举放置的总棋子数 K，将边 $(col[i], row[i])$ 的上界改为 $\lfloor Ka/b \rfloor$。

7.5 E. Coffee Central

7.5.1 题意

$N \times M$ 的网格的格点上有 K 个咖啡馆。T 组询问，每次询问给定 P_i，求出一个点使得距离其曼哈顿距离不超过 P_i 的咖啡馆数量最大。

$N, M \leq 1000, K \leq 5 \times 10^5, T \leq 10$。

7.5.2 题解

注意到将每个格点 (x, y) 的坐标变换为 $(x + y, x - y)$ 之后，到每个点曼哈顿距离不超过 P 的点构成一个正方形。设 $A[x][y]$ 表示点 (x, y) 是否有咖啡馆，在 A 变换得到的矩阵中处理二维前缀和之后，我们可以枚举答案 $O(N^2)$ 处理每个询问。

算法复杂度 $O(N^2T)$。

7.6 F. Machine Works

7.6.1 题意

有 N 台机器，购入第 i 台机器后每天可以获得 P_i 的利润。第 i 台机器售价为 P_i 元，只在第 D_i 天出售；购买之后，从第 $D_i + 1$ 天开始生产。机器可以在任意时间售出，机器 i 售出可以得到 R_i 元；若在第 p 天售出，机器无法在第 p 天生产。任意时间最多只能持有 1 台机器，开始时你没有任何机器。给定开始时你的钱数 C，确定一种方案使其在 D 天内获得的总利润最大。

$N \leq 10^5$；给出方案需满足任意时刻金钱数必须非负。

7.6.2 题解

将机器按 D 排序，设 $f[i]$ 为购买机器 i 之后，在 D_i 天可以获得的最大利润。于是 $f_0 = C, D_0 = P_0 = R_0 = 0$，

$$f_i = \max(f_j + (D_i - D_j - 1)P_j + R_j) - D_i$$

（特殊处理 $f_i < 0$ 的情况）。

整理得

$$f_i = \max((f_j - (D_j + 1)P_j + R_j) + D_iP_j) - D_i = \max(A_j + D_iB_j) - D_i$$

。将 $P_j(B_j, A_j)$ 标在坐标系中，可以发现 max 部分的计算相当于找到斜率为 $-D_i$ 并过某个 P_j 的直线中截距最大的一条。由此可得可能成为最优决策的 j（对应的点）构成凸壳，我们只需要动态维护这一凸壳。

以横坐标为序维护凸壳。需要支持的操作分为两种：计算 f_i 时，需要查询斜率为 $-D_i$ 且过某点的有最大截距的直线，这可以用三分法得到[9]；算出 f_i 后需要加点 (B_i, A_i) 并维护点集凸性，找到横坐标小于 P_i 的最后一个点以及纵坐标大于 P_i 的第一个点，分别沿两个方向进行 Graham 扫描即可。注意 P_i 可能不需加入点集。

[9]本题中查询斜率单减，也可以把平衡树当作单调队列维护。

算法复杂度 $O(N \log N)$。

7.7 G. Magic Sticks

7.7.1 题意

给定 N 条线段 $S_1..S_N$，求将它们顺序连接之后能够围成的最大面积。可以围成一个或多个多边形，且可以有线段不在多边形中。线段仅能在端点处相交。

$N \leq 500$。

7.7.2 题解

注意到三个性质：

- 最优方案中，每个多边形均被拉伸到面积最大的状态，而不会造成重叠。
- 每个多边形面积最大的形态均为其成为圆内接多边形的形态。
- 如果一段折线不连成一个多边形，其中最长的折线必然不需要使用。

由此可以动态规划。$f(i,j)$ 表示用线段 $i..j$ 可以围成的最大面积，于是 $f(1,N)$ 即为所求。转移方程为

$$f(i,j) = \max(W(i,j), f(i,l-1) + f(l+1,j)) \ (j-i>1)$$

，其中 l 为使 S_l 在 $S_i..S_j$ 中最大的一个 l，$W(i,j)$ 为 $S_i..S_j$ 可以围成的最大面积。

考虑 $W(i,j)$ 的计算。分别考虑两种情况：外接圆圆心在多边形之内和之外。两种情况下，多边形各边在半径为 r 的圆上对应的弧的圆心角之和均关于 r 单调[10]，所以可以二分计算，时间复杂度为 $O((j-i)\log N)$。

考虑动态规划的复杂度。计算 $f(i,j)$ 的复杂度

$$T(i,j) = T(i,l-1) + T(l+1,r) + O((j-i)\log N)$$

，于是最坏情况下 $T(1,n) = O(N^2 \log N)$。

7.8 H. Mining Your Own Business

7.8.1 题意

给定无向图 G，选择最少的点给它们做上标记，使得任意一个点被删除后，图中每个联通分量内至少有一个带标记的点。并求出标记方案数。

边数 $M \leq 5 \times 10^4$。

7.8.2 题解

考虑原图是一棵树的情况。很明显我们需要且仅需要标记每个叶子。

一般情况下，求出原图的所有点双联通分量。如果双联通分量数超过 1，我们需要在“叶子双联通分量”（仅与一个割点相邻的双联通分量）上各标记一个点；否则，标记任意两个点。

复杂度 $O(N+M)$。

[10]圆心在多边形外时，最长的边对应着优弧

7.9 I. Mummy Madness

7.9.1 题意

一个无限正方形网络中有 N 个木乃伊 $M_1..M_n$ 和一个人 P，木乃伊和人都在站在某个网格里。人和木乃伊轮流移动（你走第一步），每一步中人可以移动到 8 个相邻网格中的某一个，木乃伊则会移动到相邻网格中，与人的位置的欧式距离最近的一个。

判断人是否可能逃脱；如果不能，计算最长的不被木乃伊抓到的时间。

$N \leq 10^5$。

7.9.2 题解

不考虑人能逃脱的情况，二分最长的存活时间。此后问题转化为对于给定的 T，是否存在一种可以存活 T 单位时间的方案。

注意此时我们可以把移动规则改成“人先移动 T 步，木乃伊再移动 T 步”[11] 。所以问题等价于，人的可达范围内是否存在一点不被任意一个木乃伊的可达范围包含。

人和木乃伊的可达范围均为 $(2T+1) \times (2T+1)$ 的正方形。问题转化为矩形面积并，可以用扫描线解决。复杂度 $O(N \log N \log W)$。

还有一个问题：二分上界的选择与“能逃脱”情况的判断[12]。注意到假如人被抓到，他必然会在 $W = \sum(dis(M_i, P))$ 的时间以内被抓到（其中 dis 为曼哈顿距离），且这一上界是上确界。由此二分上界可以设为 W；如果人能维持 $W+1$ 单位时间，则人一定可以逃脱。

7.10 J. Pyramids

7.10.1 题意

用 N 块石头搭若干金字塔，满足

- 所有石块都被使用；
- 金字塔数尽量少；
- 所有金字塔两两不同；
- 将所有金字塔包含的石块数排序得到的列表的字典序最小。

金字塔分为两种：高金字塔和矮金字塔。底座大小为 i $(i > 1)$ 的高金字塔需要 $\sum_{1 \leq j \leq i} j^2$ 块石块；底座大小为 i $(i > 2)$ 的矮金字塔需要 $\sum_{1 \leq j \leq i} j^2[j \equiv i \ (mod\ 2)]$ 块石块。

$N \leq 10^6$。

7.10.2 题解

不考虑字典序限制，我们可以动态规划，复杂度 $O(N^{4/3})$。

为了使方案字典序最小，注意到有解时金字塔数不超过 6，所以压位保存整条路径。

[11] 由于相邻的定义是 8 联通的，我们可以分别考虑 x y 两方向上的运动。

[12] 它们是同一个问题 /w\

7.11 K. Trash Removal

7.11.1 题意

给定简单多边形 P，计算其宽度[13]。

点数 $N \leq 100$。

7.11.2 题解

$O(N^2)$ 做法可以枚举 P 与平行线贴合的一条边，计算平行线间的最小距离。

$O(N \log N)$ 的做法是旋转卡壳 [14]。

8 2012 Warsaw

8.1 A. Asteroid Rangers

8.1.1 题意

三维空间中有 N 个点分别以不同速度作匀速直线运动，任意两个点之间可以连边，边权为其欧式距离。

给定每个点的初始位置和速度，求从初始时刻开始，最小生成树变化的最少次数。

$N \leq 50$。数据保证初始时刻 MST 唯一，且任意在 t 时刻成为最优的生成树在 $t + 10^{-6}$ 时间以内是唯一的最小生成树。

8.1.2 题解

只有在两条边的大小关系发生变化时，MST 才有可能变化。所以找到所有两条边的大小关系发生变化的时间点，验证 MST 是否改变即可。

可以利用生成树的环切性质完成这一判断。具体地，如果从某一时刻开始边 (u, v) 的权值小于 (w, x)，那么 MST 改变等价于 (w, x) 在之前的最小生成树中，且在生成树中连接 u v 的路径上。

总共有 N^4 个需要验证的时刻，每个时刻验证生成树是否发生变化的复杂度为 $O(N)$，由此算法的最坏复杂度为 $O(N^5)$ 。不过注意到平均情况下，算法检查的 (w, x) 很少在生成树上，所以 $O(1)$ 实现这一判断就足以通过本题数据。

8.2 B. Curvy Little Bottles

8.2.1 题意

一个瓶子的表面可以看做多项式曲线 $f(x)(x_l \leq x \leq x_r)$ 绕 X 轴旋转，再加上瓶底得到的。

给定瓶子的描述和整数 D，求刻度线 $kD(k \in N)$ 在瓶上的高度。多项式次数 $N \leq 10$。

8.2.2 题解

瓶子的容量是关于高度的多项式函数。积分求出该函数，之后二分求根即可。

复杂度 $O(N^2 logW)$。

[13] P 的宽度的定义为最小的 W，使得 P 旋转之后可以与距离为 W 的平行线不相交。

[14] Check here

8.3 C. Bus Tour

8.3.1 题意

$N+2$ 个城市通过双向道路连接。求一条尽可能短的路线，使其

- 从 0 出发，按任意顺序经过 $1..N$ 并分别停车（可以中途经过某个城市而不停车）到达 $N+1$，然后经过 $1..N$ 分别停车，回到 0。
- 从 0 到 $N+1$ 的路上前 $\lfloor N/2 \rfloor$ 个停车的城市在从 $N+1$ 回到 0 途中也应当先停，两趟中停车顺序可以不同。

$N \leq 20$。

8.3.2 题解

动态规划预处理 $f(S), g(S)$ 分别表示从 0 或 $N+1$ 出发，经过子集 S 中的城市到达 $N+1$ 或 0 的最短路，之后枚举前 $\lfloor N/2 \rfloor$ 个停车的城市集合 S，更新答案。

算法复杂度 $O(N2^N)$。

8.4 D. Fibonacci Words

8.4.1 题意

Fibonacci 字符串的定义如下：

$$\begin{aligned} F(0) &= \text{“0”} \\ F(1) &= \text{“1”} \\ F(n) &= F(n-1) + F(n-2) \end{aligned}$$

给定 p, n，计算字符串 p 在 $F(n)$ 中出现的次数。

$n \leq 20, len(p) \leq 100$。

8.4.2 题解

设 $g(i)$ 为 p 在 $F(i)$ 中出现的次数，$p(i,j)$ 为 p 长度为 j 的前缀是否是 $F(i)$ 的后缀，$s(i,j)$ 为 p 长度为 j 的后缀是否为 $F(i)$ 的前缀。于是

$$g(i) = g(i-1) + g(i-2) + \sum_j (p(i-1,j) * s(i-2, len(p)-j)) \ \ (len(F(i-2)) \geq len(p))$$

p, s 和较小的 $g(i)$ 的值可以用字符串哈希快速得到。算法复杂度 $O(len(p)n)$。

8.5 E. Infiltration

8.5.1 题意

给定竞赛图 G，选择最少的点使得图中的每个点或者被选择，或者是某个被选择的点的后继。

点数 $n \leq 75$。

8.5.2 题解

考虑这样的一种贪心：每次选择出度最大的点，删除它和它所有的后继。由于每次至少会删除 $[\frac{N+3}{2}]$ 个点，对于 $N = 75$ 的情况我们可以得到大小不超过 6 的答案。

为了确定是否有更优的答案存在，我们只需要检查 $\sum_{1\leq i<6}\binom{n}{i} < 2\times 10^7$ 种方案。使用位压缩可以在 $O(1)$ 时间内判断每种方案是否合法。

8.6 F. Minimum Cost Flow

8.6.1 题意

三维空间中有 N 个水管节点和 M 条水管。水从名为源点的节点流入水管网，从名为汇点的节点流出。源点的水压可以调节，使得高度低于某个 K 的所有与源点联通的节点充水。

一些节点上有开孔，有开孔的节点不能充水。为了堵上开孔，可以在两个有开孔的节点之间新建管道，费用为管道长度；也可以花费 0.5 的费用堵住一个开孔。

计算为了让汇点充水需要的最小花费。

$N \leq 400, M \leq 5\times 10^4$。

节点可视为点；不需考虑水管相交的情况；节点坐标均为互异整数。

8.6.2 题解

枚举充水节点的最大高度 H，删去高度高于 H 的点。对于能用原管道相连的节点，其中只要有一个节点充水其它的必然都充水，所以将它们缩成一个点，开孔数为其开孔数之和。重建缩点之后的图中的边。

在新图中，问题转化为选择一些点和边，使得源点所在点和汇点所在点联通。选择一个点的代价为其开孔数乘 0.5（堵住这个点所有开孔的代价），选择一条边的代价为其权值减 1（选择一条边减少了堵住两个开孔的代价）。

由于节点坐标均为整数，新图中边权非负，所以最终方案是新图中的一条从源到汇的简单路。为了保证管道可以建成，路径应满足除源汇所在点之外，路径中顶点的开孔数不小于 2；源汇所在点的开孔数不小于 1。由此，在满足条件的点中求最短路即可。

复杂度 $O(N(N+M)logM)$。

8.7 I. A Safe Bet

8.7.1 题意

$R\times C$ 的矩形网格中有 M 面镜子，镜子沿格子的某一对角线方向放置。如果一个镜子的放置方案满足从左上角横向射入网格的激光可以经过反射从右下角横向射出，我们称这一放置方案是合法的。如果一个镜子的放置方案满足恰插入一面镜子后成为合法方案，我们称这一放置方案是安全的。给定一个镜子的放置方案，判断其是否安全。如果安全，输出新插入镜子字典序最小的位置；否则，判断其是否合法。

镜子的两面均可反射；同一个网格中只能有一面镜子。

$R, C \leq 10^6, M \leq 4\times 10^5$。

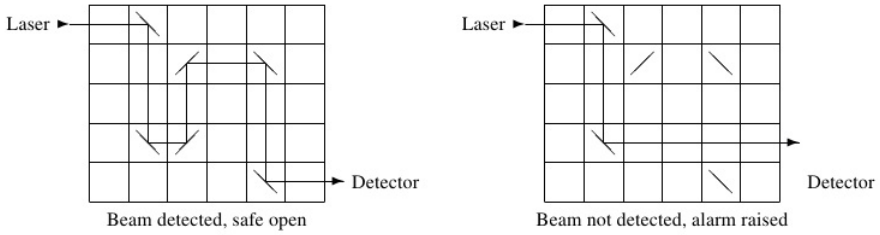

Figure 3: 一种镜子的放置方案。

8.7.2 题解

判断一个方案是否合法只需要模拟光线的行进路线。用平衡树维护横向和纵向镜子的位置，模拟的复杂度为 $O(MlogM)$。

为了判断一个方案是否安全，我们计算从左上角射入光线的行进路线和从右下角射入光线的行进路线，然后判断它们是否有交点。这一判断可以用扫描线算法完成，复杂度 $O(MlogM)$。

8.8 K. Stacking Plates

8.8.1 题意

给定 N 个有序数组，用以下两种操作将它们合并。

- 拆分：将一个数组从某个位置分割。
- 合并：如果两个数组满足其中一个数组的最大值不大于另一个的最小值，可以将它们顺序连接。

求最少操作次数。

$N \leq 50$，输入数组长度 $H \leq 50$。

8.8.2 题解

显然最小化操作次数等价于最小化分割次数。

将数组元素离散化，删除每个数组中的重复项。如果输入中的数两两互异，我们只需要将每个数组分割为 $[l_1, r_1], [l_2, r_2], ..[l_k, r_k]$ 这样的若干段；否则，我们需要考虑在合并之后序列中，同一个数在不同数组中的多次出现之间的相对顺序。

从小到大考虑每个数（在不同数组中对应的元素）的顺序安排，显然只能有一个元素有可能与它所在数组中的上一项相连，只能有一个元素可能与它所在数组中的下一项相连，而只有相连才能减少分割次数。由此我们可以动态规划。设 $f(i, j)$ 表示大小不超过 i 的数在合并后序列中的顺序均已确定，合并后的最后一项本来在数组 j 中时，当前已经分割的最少段数，于是 $2min(f(maxv, i)) - N - 1$ 即为所求。转移规则如下：

- 如果数组 j 中不包含 $i+1$，用 $f(i, j) + count(i+1)$ 更新所有的 $f(i+1, j')$，其中 $count(i)$ 表示包含 i 的数组个数。
- 否则，用 $f(i, j)+count(i+1)-1$ 更新所有的 $f(i+1, j')$，用 $f(i, j)+count(i+1)-[count(i+1) = 1]$ 更新 $f(i+1, j)$。

算法复杂度 $O(N^3H)$。

8.9 L Takeover Wars

8.9.1 题意

考虑两个公司 X 和 Y 之间的商业战争。每个公司包含一系列子公司，每个子公司的市场价值是已知的。

每个子公司可以同一公司中的另一个子公司合并（Merge），合并后新公司的价值等于原先公司的价值之和；也可以收购（Takeover）另一个公司中价值小于自身价值的某个子公司，收购后该子公司价值不变，被收购子公司消失。

如果一个公司中所有子公司均被收购，它就输掉了这场战争。

两个公司轮流行动，X 先行；每一步当前操作的公司可以进行一次合并，收购或者什么都不做。判断在双方均使用最优策略时获胜的公司。

两公司子公司数 $n, m \leq 10^5$；输入保证不会出现平局情况。

8.9.2 题解

经过分析我们可以得到以下性质。

定理 8.1. *最优决策中，以下性质成立。*

a) 如果上一步中对方合并了最大的两个子公司，且这一步中我方可以并购对方最大的子公司，则我方必胜。

b) 如果上一步中对方并购了我方最大的子公司，这一步中我方只能合并最大的两个子公司。

c) 我方仅在可以并购对方最大子公司的情况下发起并购。

d) 我方如果合并，只会合并最大的两个子公司。

证明. 以下设我方子公司按价值降序排列为 $A[1..n]$，对方子公司排列为 $B[1..m]$。

考虑游戏的游戏图[15]。很明显这是一个无环图，所以我们按照拓扑顺序归纳证明这四个结论。

对于没有出边的节点，结论是显然的。

假设对节点 v 的所有后继，结论 a - d 成立，我们要证明对于节点 v 结论成立。

a) 由于 $A[1] > B[1] \geq B[2] + B[3]$，我方并购对方子公司 $B[1]$ 后，对方无法并购 $A[1]$。所以在之后的局面中，我方总能并购对方最大的子公司，因而我方必胜。

c) 很明显并购时我方总会并购可以并购的最大公司。假设在局面 v 我方无法并购对方的最子大公司，然而我方对另一子公司发起并购并最终获胜。我们要证明，不进行这次并购也可以获胜。

此时 $B[1] > A[1]$。考虑之后的情况。

(a) 由于性质 c 在后继局面中成立，下一回合对方将会并购我方的最大子公司；

(b) 由于 b 在后继局面中成立，我方在下一回合将合并最大的子公司；

[15] 对于每个局面建一个点；如果局面 x 可以在一步内变换为局面 y，连边 $x \to y$

(c) 由于 a 在后继局面中成立且我方获胜，对方在之后第二回合里将会合并最大的子公司；

(d) 如果之后第二回合时我方可以并购对方最大的子公司，根据 a 我方获胜；否则，根据 a c，双方只能不断合并，直到在之后第 p 回合中，我方能够并购对方的最大公司时，我方获胜。

(e) 此时有 $A[2..p+1] > B[1..p]$[16]。由此 $A[1..p] > B[1..p]$，所以如果我方在这一回合（局面 v）合并最大的两个子公司，按照以上方法推演，双方会不断合并，直到在第 $p' \le p$ 回合我方获胜。[17]

d) 假设在局面 v，我方合并了 $A[i]$ 和 $A[j]$（$j+i>3$）并获胜。我们要证明合并 $A[1]$ 和 $A[2]$，或者放弃这一步也能获胜。

(a) 如果在下一回合中对方发起并购，根据 c 对方会并购 $A'[1]$。设我方在之后的第 p 回合获胜，类比 c 可得 $A'[2..p+1] > B[1..p]$，于是将 $A[i]+A[j]$ 换为 $A[1]+A[2]$ 之后，不等式左边不减，我方仍然可以获胜。

(b) 如果在下一回合中对方合并，同理 $A'[1..p] > B[1..p+1]$，替换 $A[i]+A[j]$ 后我方仍可获胜。

b) 由于 c 在局面 v 成立，我方不会发起并购。由于 d 在局面 v 成立，我方会合并最大的子公司。

□

考虑直接搜索游戏过程，每次优先尝试并购，只在并购无法获胜时考虑合并。可以发现，搜索树的形态如下：

```
开始 - 并购 - 合并 - 合并 - .... - 合并 - 并购（胜负确定）
    \- 合并 - 合并 - 合并 - .... - 合并 - 并购（胜负确定）
```

由此，算法复杂度为 $O(N+M)$。

9 2013 Saint Petersburg

9.1 A. Self-Assembly

9.1.1 题意

有 N 种可视为正方形的分子，每种分子的每一条边有一个标识符，标识符为 “A+ .. Z+”，“A- .. Z-”，“00” 中的一种。如果两个分子的两条边的标识符分别为 “c+” 和 “c-”（c 为任意字母），它们可以通过这两条边连接。

给定每种分子的标识符，判断是否用它们可以在平面上拼出无限大的结构体。每种分子的数量不限，分子可以旋转或翻转。

$N \le 4 \times 10^4$。

[16]指 $\sum_{i=2}^{p+1} A[i] > \sum_{i=1}^{p} B[i]$，以下类似

[17]作为严密的证明，这里（以及之后 d 中的部分）还需要说明改变策略后不会提前失败。这是可以类似推出的。

9.1.2 题解

首先结构体必须在平面上的限制是不用考虑的，因为我们可以通过旋转和翻转，将一个无限长的链转化为只向上或向右延伸的形态。

问题转化为，是否存在一个环，使得环上相邻分子可以连接。将分子看做点构图找环只能得到 $O(N^2)$ 的算法；对每种标识符建立一个点，将分子当作连接标识符的边，在这个图中找环，可以得到复杂度为 $O(N)$ 的算法。具体地，对于某种分子中两个不同位置的标识符 a，b，连边 $a \to dual(b)$，其中 $dual(b)$ 表示可以与 b 连接的标识符。

9.2 B. Hey, Better Bettor

9.2.1 题意

考虑一种赌博游戏：游戏可以进行任意多局，在每局中你需要支付 1 元，有 p 的概率获胜，如果该局获胜，可以得到 2 元。在任意时刻你都可以退出，如果退出时收益为 M 元，你会额外得到 $\max(0, -xM)$ 元，其中 x 为给定常数。

给定 x, p，计算采用最优策略时的期望收益。$0 \le x < 1, 0 \le p < 0.5$，且 x 和 p 是 0.0001 的倍数。

9.2.2 题解

在一个策略中，我们需要决定何时退出。很明显，退出条件只与当前收益有关，所以任意策略都可以这样描述：在收益不小于 A 元或不大于 B 元时退出游戏（$B < 0 < A$）。

如果 A, B 已经确定，为了计算期望收益我们只需计算收益 A 元的概率。设 $P(i)$ 为当前收益 i 元时，能在收益 A 元时退出游戏的概率。于是

$$P(i) = P(i+1)p + P(i-1)(1-p)$$

，亦即

$$P(i+1) = \frac{1}{p}P(i) - \frac{1-p}{p}P(i-1)$$

。代入 $P(B) = 0, P(A) = 1$ 可得

$$P(i) = \frac{\lambda^i - \lambda^B}{\lambda^A - \lambda^B}$$

，其中 $\lambda = \frac{1-p}{p}$。于是收益 A 元的概率 $P(0) = \frac{1-\lambda^B}{\lambda^A - \lambda^B}$，期望收益 $W(A, B) = P(0)A + (1-x)(1-P(0))B$。

考察 W 函数。对 A 求偏导，得

$$\frac{\partial W}{\partial A} = \frac{-1+\lambda^B}{(\lambda^B - \lambda^A)^2}(\lambda^B - \lambda^A + \lambda^A(-1+x+A)\ln\lambda) = \frac{-1+\lambda^B}{(\lambda^B - \lambda^A)^2}M$$

。$A > 0$ 时，M 的系数部分恒负。而

$$\frac{\partial M}{\partial A} = \lambda^A \ln^2(\lambda)(x + A - 1) > 0$$

，且

$$M|_{A=0} = \lambda^B - 1 + (-1+x)\ln\lambda < 0$$

，所以 W 是关于 A 的单峰函数。

对 B 求偏导，得

$$\frac{\partial W}{\partial B}=-\frac{(\lambda^A-1)}{(\lambda^A-\lambda^B)^2}(\lambda^B\ln\lambda(A+B(x-1))+(x-1)(\lambda^A-\lambda^B))=-\frac{(\lambda^A-1)}{(\lambda^A-\lambda^B)^2}N$$

。$B<0$ 时，N 的系数部分恒负；而

$$\frac{\partial N}{\partial B}=\lambda^B\ln^2\lambda(A+B(x-1))>0$$

，且

$$N|_{A+B(x-1)=0}=(x-1)(\lambda^A-\lambda^B)<0$$

，所以 W 是关于 B 的单峰函数。

由此，$f(B)=\max_A W(A,B)$ 也是单峰函数，所以我们可以嵌套三分得到答案。经过简单验证，$x=0.4999,p=0.9999$ 时三分范围最大，此时 $B=20528,A=2498$。取这两个数为三分上界，得到 $O(\log A\log B)$ 算法。

9.3 C. Surely You Congest

9.3.1 题意

有向图的顶点上有 N 个人，所有人同时出发，沿到 t 的最短路行进。两个人不能同时进入一条边，但可以同时经过一个点。给定每个人的出发点，确定最少去掉多少人可以让剩下的人到达 t。

$N\le 25000,M\le 50000$。

9.3.2 题解

如果所有人的出发点到 t 的最短路等长，构造到 t 的最短路径 DAG[18]之后，问题可以用网络流解决：DAG 上的边容量设为 1，从源点到每个人的出发点连边容量为 1，求源点到 t 的最大流。

注意如果两个人的出发点到 t 的最短路长度不同，这两个人不可能同时进入一条边，亦即他们是互不影响的。所以将所有人按照到 t 距离分组，分别网络流计算即可。

9.4 D. Factors

9.4.1 题意

设 $f(N)$ 为 N 的质因子分解的排列方案数，例如 $f(20)=3$。给定 k，求最小的 N 使得 $f(N)=k$。

$k\le 2^{63}$；数据保证有解且最小的 N 不超过 2^{63}。多组数据捆绑。

9.4.2 题解

设 $N=2^{k_1}3^{k_2}5^{k_3}...$，由于 N 是最小的答案，$k_i\ge k_{i+1}$。注意到 2^{63} 范围内满足这一条件的 N 只有 43606 个，所以生成所有这样的 N，平衡树维护 $f(N)$ 即可。

[18]构造方法：在图中保留满足 $dis(u,t)+w=dis(v,t)$ 的所有边 (u,v,w)。

9.5 E. Harvard

9.5.1 题意

考虑这样的一种计算机：数据储存在 B 个等大内存库中，内存库 0 有专用指令访问；如果要访问内存库 $i(i \neq 0)$，需要提前用一条指令将寄存器 A 的值设置为 i，之后用一条指令获取相应数据。

给定一段仅包含变量访问操作和固定次数循环的程序，将每个变量分配到内存库中的一个位置，使得用于内存访问的指令数最小。

每个内存库中的位置数不超过 13；变量数不超过 13。

9.5.2 题解

搜索分配到每个内存库的变量。添加以下剪枝足以通过所有测试：

- 将尽量多的变量分配到内存库 0。
- 完成库 0 的分配之后，遍历输入程序，计算 $B[i][j]$ 表示去掉已被分配的变量之后，变量 i 恰在变量 j 之后被访问的次数。
- 除库 0 之外，其它的所有库是等价的。因此不妨令库 i 中标号最小的变量小于 $i+1$ 中最小的变量。
- 如果当前指令数加上未分配变量的出现次数不小于答案，中止当前状态的搜索。

9.6 F. Low Power

9.6.1 题意

有 N 个机器，每个机器上有 2 个芯片，每个芯片可以放 k 个电池。每个芯片的能量是其中电池的能量的最小值。一个机器的两个芯片的能量差值越小，这个机器工作得越好。

现有 $2Nk$ 个电池，已知它们的能量，我们要把它们分配给每个机器的芯片，使得所有机器的芯片能量差的最大值最小。

$2Nk \leq 10^6$。

9.6.2 题解

考虑在二分答案之后，分配每个机器的两个芯片上能量最小的电池。显然这两块电池的能量在所有电池中是最接近的。设排序之后电池的能量数组为 $A_1..A_{2Nk}$，当前二分的答案为 A，每个机器中能量最小的电池编号为 M_i，于是 M 应当满足

$$
\begin{aligned}
&M_1 = 1\\
&A_{M_i+1} - A_{M_i} \leq A\\
&M_{i-1} + 1 < M_i \leq 2ik + 1
\end{aligned}
$$

（最后一式是剩下的电池可以分配的充要条件。）由上可知，我们可以贪心地让 M_i 尽量小，所以 $O(N)$ 扫描即可。

算法复杂度 $O(N \log N)$。

9.7 H. Matryoshka

9.7.1 题意

N 个套娃排成一行，用最少的步数将它们组装成若干个完好的套娃集（套娃集的所有套娃的大小恰为 $1..M$，其中 M 为任意值）。

组装过程中你只可以合并两个相邻的套娃集，所用步数等于被打开过的套娃总数。例如，合并 $[1,2,5]$ 和 $[3,4]$ 的代价为 3，合并 $[1,2]$ 和 $[5]$ 的代价为 1。

$N \leq 500$，套娃大小 $K \leq 500$。

9.7.2 题解

动态规划：设 $f[i][j]$ 为将套娃 $i..j$ 合并成一个集合的最小代价，$g[i]$ 为将前 i 个套娃合并为若干集合的最小代价，于是 $g[i] = \min(g[j] + f[j+1][i] : \text{套娃 j+1..i 构成一个完好的集合})$，$g[N]$ 即为所求。

考虑 f 的计算。很明显 $f[i][j] = \min(f[i][i'] + f[i'+1][j] + W(i, i', j))$，其中 $W(l, m, r)$ 表示将套娃集 $[l, m]$ 和 $(m, r]$ 合并的代价。注意到在合并两个集合的过程中，只有最小的 k 个套在一起的套娃不会被拆开，所以我们可以在 $O(1)$ 时间内计算 W，因此算法复杂度为 $O(N^3)$。

9.8 I. Pirate Chest

9.8.1 题意

有一个池塘，最大水深为 D。池塘底部可以划分为 $N \times M$ 个正方形网格，网格 (i, j) 的高度为 $H[i][j]$。求出体积最大的长方体，满足长方体的棱长为整数，一底边长不超过 A，另一底边长不超过 B，且沉入池塘后不会有暴露在水上的部分。

$N, M \leq 500, H[i][j] \leq 10^9$。

9.8.2 题解

枚举长方体左上角的位置似乎只能得到 $O(N^4)$ 的算法。考虑枚举长方体左上角所在行 i 和长方体的宽 j（占据的行数），得到数组 $F[k] = \max(H[i..i+j-1][k])$，于是如果长方体包含列 $l..r$，其沉入在水中后的高度即为 $\max(F[l..r])$。枚举 $\max(F[l..r])$ 所在的位置之后，长方体显然应当包含尽量多的列。所以预处理 $L[i] = \max(j : j < i, F[j] > F[i]) + 1, R[i] = \min(j : j > i, F[j] > F[i]) - 1$，枚举 i 更新答案。

9.9 J. Pollution Solution

9.9.1 题意

求简单多边形 P 和半圆的交集面积。

多边形点数 $N \leq 100$。

9.9.2 题解

受到简单多边形面积算法的启发，枚举多边形上的相邻点 P_i, P_{i+1}，计算 $\triangle OP_iP_{i+1}$ 与半圆的面积交 s，如果 $P_i \times P_{i+1} > 0$，则答案增加 s，否则答案减去 s。最终答案的绝对值即为所求。

复杂度 $O(N)$。

9.10 K. Up a Tree

9.10.1 题意

考虑树的前序、中序和后序遍历函数，三个函数的六次递归调用的位置已经被随机重排，其它部分保证正确。给定对于一棵树，三个修改过的函数的输出，判断重排后递归调用的顺序。

树中节点数不超过 26。

9.10.2 题解

枚举重排后的顺序，记忆化搜索验证。

注意可能有某棵子树的某个遍历结果不存在的情况，此时我们需要枚举根，或者枚举左子树的大小。