

Problem A. 疯狂星期四，V 我 50！

将所有字母转换为小写字母之后，枚举每个串，使用字符串查找函数（比如 C 中的 `strstr`，C++ 中的 `find`）或者手动找到是否存在这个串即可。若五个串都存在则是 Yes。

时间复杂度 $\mathcal{O}(|s|)$ 。

Problem B. 最大差值

设一个数的个位为第 1 位，十位为第 2 位，以此类推。

那么若将数字 x 放到 n 位数的第 i 位，对答案的贡献为 $x \times 10^{i-1}$ ；若放到 m 位数的第 i 位，对答案的贡献为 $-x \times 10^{i-1}$ 。因此只需要贪心地将大的数字放到产生贡献大的位置即可。

注意答案中一定不存在退位，故无需实现高精度减法，注意前导零的处理即可。

时间复杂度为 $\mathcal{O}(n+m)$ 。

Problem C. 小水獭的 Codeforces Rating

本题给出的计分函数

$$r + \left\lceil \frac{s_i - r}{4} \right\rceil$$

无论对 r 还是对 s 都是单调增的，因此某场比赛如果会使小水獭得分下降，那么他一定会参赛。

时间复杂度 $\mathcal{O}(n)$ 。

一个小彩蛋：本题是 BCPC2021 决赛 A 题的镜像问题。

Problem D. 泽瑞拉

因为数据范围是 1000，所以可以用动态规划来解决。具体来说， $dp_{i,j}$ 表示当前已经触发了 i 次自爆绵羊的亡语， j 次牛头人的亡语的概率。时间复杂度为 $\mathcal{O}(n^2)$ 。

当然，如果你数学比较好，也可以这么计算：

每个泥浆怪的概率是等概率的分布在 y 次自爆绵羊之间触发的，恰好在最后一次之后的概率是 $\frac{1}{y+1}$ ，那么总共期望即为 $\frac{x}{y+1}$ 。时间复杂度为 $\mathcal{O}(1)$ 。

Problem E. 炸金银花

首先判断给定牌组的牌型，然后计算每个不大于当前牌型的牌型中有多少个牌组小于当前牌组即可。

以单牌为例，如果当前牌型为单牌且数值分别为 a, b, c ($a > b > c$)，小于该牌组的牌组个数为：

$$[(\sum_{i=1}^{a-1} (\sum_{j=1}^{i-1} (j-1)) - 1) + \sum_{i=1}^{b-1} (i-1) + (c-1)] \times (m^3 - m)$$

左边中括号中第一项为不考虑花色时 $a_0 < a$ 的单牌 (a_0, b_0, c_0) 个数，第二项为 $a_0 = a, b_0 < b$ 的个数，第三项为 $a_0 = a, b_0 = b, c_0 < c$ 的个数。需要注意去除掉顺子的情况。

右边小括号中是花色的种类数。需要注意去除掉同花的情况。

其他牌型的情况同理。

利用公式 $\sum_{i=1}^n i^2 = n \times (n+1) \times (2n+1)/6$ 和 $\sum_{i=1}^n i = n \times (n+1)/2$ 即可 $\mathcal{O}(1)$ 计算出结果。

Problem F. Toxel 与 Villages: Landcircles

首先说明答案是 $(2n-5)!!$ ，即 $1 \times 3 \times \cdots \times (2n-5)$ 。经过 $\mathcal{O}(n)$ 预处理后，可在 $\mathcal{O}(1)$ 的时间内回答每组询问。

设共有 m 个额外点，则该树的总度数为 $n+3m$ 。又因为总点数为 $n+m$ ，可得 $n+3m = 2(n+m)-2$ ，即 $m = n-2$ 。

现在考虑对问题进行一个形式化的定义。对所有 $2n-2$ 个点，带标号的树，定义两棵树 T, T' 等价，当且仅当存在一个将 T 的节点映射到 T' 的节点的一一对应 σ ，使得对 $\forall i \in 1 \dots n$ ，有 $\sigma(i) = i$ ，且 T 中任意两个节点 u 和 v 相连接，当且仅当 T' 中对应的两个节点 $\sigma(u)$ 和 $\sigma(v)$ 相连接。求等价类的数量。

对于某棵树 T ，与其等价的树的数量至多为 $(n-2)!$ 种，因为不同的 σ 只有这么多。下面证明每个等价类的大小恰为 $(n-2)!$ 。当 σ 不为单位排列时，考虑不停地删除 T 中的某个叶子，直到遇到某个 $\sigma(u) \neq u$ 的节点 u 。 u 一定大于 n ，其在 T 中度为 3。因此在 T, T' 中， u 有两个邻点相同，一个邻点不同。故 T 一定不等于 T' 。接下来可以简单推出对 T 应用不同的 σ 变换所得的树各不相同。因此每个等价类的大小恰为 $(n-2)!$ 。

所有 $2n-2$ 个点，带标号，且度数满足条件的树可用 Prüfer 序列计数。这样的树对应了 $n+1 \sim 2n-2$ 各出现两次的序列，其数量为 $\frac{(2n-4)!}{(2!)^{n-2}}$ 。将其除以 $(n-2)!$ 即可得出答案。

另一种解法是递推。考虑从 $n-1$ 的答案转移到 n 的答案，可任选一条边，在边上加一个额外点，再将 n 连到该额外点上。由于 $n-1$ 时共有 $2n-5$ 条边，因此可立即得到结论。这里就不严格证明正确性了。

Problem G. 小 U 的矩形数量

首先考虑两个矩形包含的情况，相当于在每一维都选择四个点，内部两个给一个矩形，外部两个给另一个矩形，方案数 $\binom{n}{4} \times \binom{m}{4}$ 。

然后我们发现两个矩形相交的充要条件是，这两个矩形在每一维都相交。

考虑容斥，先选定两个矩形在 x 轴上不相交，然后在 y 轴上的点随便选，方案数 $\binom{n}{4} \times \binom{m}{2} \times \binom{m}{2}$ 。

在 y 轴不相交的方案数同理， $\binom{m}{4} \times \binom{n}{2} \times \binom{n}{2}$ 。在两维都不相交的矩形被我们重复计算了，要减去 $2 \times \binom{n}{4} \times \binom{m}{4}$ 。

总方案数， $\binom{n}{4} \times \binom{m}{2} \times \binom{m}{2} + \binom{m}{4} \times \binom{n}{2} \times \binom{n}{2} - \binom{n}{4} \times \binom{m}{4}$ 。

时间复杂度为 $\mathcal{O}(1)$ 。

Problem H. Toxel 与宝可梦对战特训

不妨先将输入从小到大排序，其次将题目中的排列分解为若干长度不小于 2 的环，从环的角度来考虑问题。结论如下：当 n 为偶数时，所有 i 与 $n+1-i$ 配对结合为二元环。 n 为奇数时，令 $m = \frac{n+1}{2}$ ，则 $m-1, m, m+1$ 配对结合成三元环，其它元素的结合方式与偶数相同。

本题的证明较为复杂，主要思想为调整法。首先可以证明环长不超过 3，考虑证明长度至少为 4 的环都能拆成更小的环而使答案不劣。对于一个这样的环，考虑其权值最大和最小的元素 $a_{\min}, a_{\max}$ 。分三种

情况讨论：

- 若环的结构为 $\min \rightarrow \max \rightarrow p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_x \rightarrow \min$ ，即最大值为最小值的后继，那么将环拆成 $\min \rightarrow \max \rightarrow \min$ 和 $p_1 \rightarrow \cdots \rightarrow p_x \rightarrow p_1$ 。考虑后者增加的乘积对，仅有 $a_{p_1}a_{p_x}$ ，它显然不大于 $a_{p_1}a_{\max}$ 。
- 若环的结构为 $\min \rightarrow p_1 \rightarrow \max \rightarrow p_2 \rightarrow \cdots \rightarrow p_x \rightarrow \min$ ，即最小值在环上走两步到最大值，那么将环拆成 $\max \rightarrow p_1 \rightarrow \max$ 和 $\min \rightarrow p_2 \rightarrow \cdots \rightarrow p_x \rightarrow \min$ 。考虑后者增加的乘积对，仅有 $a_{\min}a_{p_2}$ ，它显然不大于 $a_{\max}a_{p_2}$ 。
- 若环的结构为 $\min \rightarrow p_1 \rightarrow \cdots \rightarrow p_x \rightarrow \max \rightarrow p_{x+1} \rightarrow \cdots \rightarrow p_y \rightarrow \min$ ，即最小值在环上走至少三步到最大值，那么将环拆成 $\min \rightarrow p_1 \rightarrow \cdots \rightarrow p_x \rightarrow \max \rightarrow \min$ 和 $p_{x+1} \rightarrow \cdots \rightarrow p_y \rightarrow p_{x+1}$ 。考虑后者增加的乘积对，有 $a_{\min}a_{\max}$ 和 $a_{p_{x+1}}a_{p_y}$ ，它们均不大于 $a_{p_{x+1}}a_{\max}$ 。

接下来还可以证明，至多有一个长度为 3 的环，其它环的长度均为 2。若有两个三元环，不妨设为 $p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow p_1$ 和 $q_1 \rightarrow q_2 \rightarrow q_3 \rightarrow q_1$ 。注意到三元环两两元素均产生贡献，因此我们总可以假定 p, q 均是递增的。那么将其拆成 $p_1 \leftrightarrow p_3$ ， $q_1 \leftrightarrow q_3$ ， $p_2 \leftrightarrow q_2$ 是不劣的，读者可自行验证。

至此，读者不难自证 n 为偶数时的性质，延续调整法的思路即可。 n 为奇数时，可证明三元环的三个元素相邻是最优的。证明需讨论非常多种情况，这里仅列举一种。不妨设有 $p_1 < p_2 < p_3 < p_4 < p_5$ 组成了环 $p_1 \rightarrow p_3 \rightarrow p_5 \rightarrow p_1$ 和 $p_2 \leftrightarrow p_4$ ，将其替换成 $p_2 \rightarrow p_3 \rightarrow p_4 \rightarrow p_2$ ， $p_1 \leftrightarrow p_5$ 。前者的最大乘积为 $a_{p_3}a_{p_5}$ ，后者的乘积为 $\max(a_{p_3}a_{p_4}, a_{p_1}a_{p_5})$ ，小于等于前者。可以证明，在任何情况下，只要三元环的元素不相邻，总能找到一种替换方式，使得该环中间属于其它环的元素数量减少，而乘积最大值不变。综上， n 为奇数的情形也可得证。

时间复杂度 $\mathcal{O}(n \log n)$ 。

Problem I. 烤土豆

最优策略下，每次捞土豆肯定是在某个土豆放下去的时候立刻捞起。即，捞土豆的时间和放土豆的时间是一样的。

因此可以记 dp_i 表示对于前 i 个土豆，最后一次捞土豆是在第 i 个土豆放下去的时刻，这种情况下最小的自闭度。 dp 的转移就是枚举上一次捞土豆的时刻，计算加上这一段土豆的自闭度，然后取最小值。

不妨令 $t_0 = 0, dp_0 = 0$ ，有转移方程：

$$dp_i = \min_{j=0}^{i-1} (dp_j + k + \sum_{l=j+1}^i (t_i - t_l))$$

把里面的求和拆出来，可以使用前缀和优化。令 $\text{sum}_i = \sum_{j=0}^i t_j$ ，转移方程转化为：

$$dp_i = \min_{j=0}^{i-1} (dp_j + k + (i - j) \times t_i - (\text{sum}_i - \text{sum}_j))$$

再推一下式子就可以使用斜率优化 dp 的方法求解，复杂度 $\mathcal{O}(n)$ 或 $\mathcal{O}(n \log n)$ 均可以通过。也可以使用李超线段树。

关于斜率优化 dp 的具体实现，可以参考 [OIWiki](#)。

关于李超线段树的具体实现，可以参考 [OIWiki](#)。

Problem J. 约会大作战

如果第 i 个男生与第 j 个女生约会，那么男生需要将其魅力值提升到至少 q_j ，需要的花费为 $\max(q_j - a_i, 0)$ ；同理，女生需要的花费为 $\max(p_i - b_j, 0)$ 。因此，第 i 个男生与第 j 个女生约会的花费为 $\max(q_j - a_i, 0) + \max(p_i - b_j, 0)$ 。

如果将 n 个男生看作二分图左部的 n 个点， n 个女生看作二分图右部的 n 个点，并在对应的点间连一条权值为约会花费的边，那么我们需要找到一组完备匹配，使得匹配中每条边的权值和最小。这可以转化为二分图最大权匹配问题，可以使用 Kuhn-Munkres 算法（可参考 [OIWiki](#)）或者用 Dijkstra 增广的费用流（可参考 [OIWiki](#)）解决，时间复杂度均为 $O(n^3)$ 。但后者的常数较大，可能无法在时间限制内通过。

Problem K. 异或最大值

这里给出两种不同的分治做法：

K.1 最值分治

对于区间 $[L, R]$ ，找到区间最大值所在的位置 p ，如果有多个最大值则任选一个。那么我们可以统计对于 $l \in [L, p], r \in [p, R]$ ，有多少个区间 $[l, r]$ 满足条件，然后再分治处理区间 $[L, p]$ 和 $(p, R]$ 即可。

我们可以枚举 $[L, p]$ 和 $[p, R]$ 中长度较小的区间，考虑该区间中每个位置作为端点的贡献，不妨假设 $p - L \leq R - p$ ，当前枚举到某个 $i \in [L, p]$ ，那么只要求满足 $j \in [p, R]$ 且 $a_i \oplus a_j > a_p$ 的 j 的个数。这可以借助可持久化 Trie 或拆询问后直接用 Trie 解决，设数的值域为 $[0, V)$ ，单词询问的复杂度为 $O(\log V)$ 。

下面来分析这种做法的时间复杂度。实际上，询问的个数为 $O(n \log n)$ 级别。考虑原序列的笛卡尔树，我们每次分治处理的区间对应着笛卡尔树上的一个结点，而枚举长度较小的区间相当于遍历总结点数较小的子树，这类似于树上启发式合并的过程，每个点只会被遍历（合并） $O(\log n)$ 次。也可以用轻重链剖分的理论来理解，树上每个点被遍历到的次数相当于这个点到根节点路径上轻边的数量，也为 $O(\log n)$ 级别。

因此总的复杂度为 $O(n \log n \log V)$ 。

K.2 普通分治

对于区间 $[L, R]$ ，令 $p = \lfloor \frac{L+R}{2} \rfloor$ ，我们可以统计对于 $l \in [L, p], r \in (p, R]$ ，有多少个区间 $[l, r]$ 满足条件，然后再分治处理区间 $[L, p]$ 和 $(p, R]$ 即可。

考虑枚举最大值所在的区间，不妨设为 $[L, p]$ ，从大到小枚举 $i \in [L, p]$ ，对于每个 i ，我们找到最大的 $k \in (p, R]$ ，使得 $\max(i, p) \geq \max(p+1, k)$ ，这里的 $\max$ 符号含义与题面中一样，表示区间最大值。与最值分治相同，只要求满足 $j \in (p, k]$ 且 $a_i \oplus a_j > \max(i, p)$ 的 j 的个数。显然， k 随着 i 的减小单调不减，于是只需要用一个指针简单维护。对于最大值在 $(p, R]$ 区间的处理也类似，只要注意两区间的最大值相等时如何处理避免重复即可。

这种做法的复杂度同样为 $\mathcal{O}(n \log n \log V)$ 。

Problem L. 逃跑路线

首先我们先考虑最简单的长度为 3 的链，如果起点为链的一端，那么我们可以不改变当前状态的情况下跳到链的另一端（在第一条边上绕行 k 圈后走到第二个点，再在第二条边上绕行 $k-1$ 圈后走到第三个点即可）。如果我们将第 i 个点的权值定义为 $val_i = b_i - a_i$ （模 k 意义下），那么我们可以发现第一个点的权值也可以通过绕行转移到第三个点上。

由此推广可知，如果图为二分图，黑白染色后，相同颜色的点可以视为等价，且权值可以相互转移，那么可以将任意二分图简化为仅有两个相连的点的图，其中这两个点的权值分别为黑点权值之和和白点权值之和（模 k 意义下）。如果两点权值相同，则两点都可作为起点，对应到原图则是全图所有点都可作为起点；如果两点权值相差 1（模 k 意义下），那么仅可以权值较小的点作为起点（特别地，如果两点权值分别为 0 和 $k-1$ ，那么权值为 $k-1$ 的点可作为起点），该点对应原图中相同颜色的点都可作为起点；否则全都无法作为起点。需要特判的是，如果 $k=2$ ，那么无论如何全图都可作为起点。

而对于一般图，全图都可作为起点。类似于二分图上的权值转移，我们也可以在一般图中将所有权值都转移到一个奇环的一个点上，然后可以通过在这个环上绕行将该点权值变成 0。

时间复杂度 $\mathcal{O}(n)$ 。

Problem M. 宝可梦与毒瘤 Toxel

有一些特殊情况可以先进行预处理。所有长度为 1 的维可以忽略掉（注意价值中的指数不可改变）。而如果只有一维长度大于 1，那么答案是 1，因为最终唯一的方案是切成若干 $1 \times 1 \times \dots \times 1$ 的巧克力块。下面使用类似 $\mathbf{m}$ 的记号表示一个 D 维向量。 n_i 表示各维的最大长度（同题面）。

现在考虑至少有 2 维长度大于 1 的情况，有递推式：

$$f(\mathbf{m}) = \sum_{1 \leq m'_1 < m_1, \dots, 1 \leq m'_D < m_D} f(\mathbf{m}') f(\mathbf{m} - \mathbf{m}') \times \prod \text{Values of removed chocolates}$$

其中，被分配出去的巧克力的价值之积可以这样计算。因为每个巧克力块的价值是一个乘积，愉悦度也是一个乘积，所以可以把每个维度的价值分别计算再乘起来即可。这里不妨先一起计算“左下”和“右上”角巧克力的价值，那么 m'_i 和 $m_i - m'_i$ 在 2^D 个巧克力块中各出现了一半，即 2^{D-1} 次。再除以两个保留下来的巧克力的价值，可得：

$$\prod_{i=1}^D (m'_i (m_i - m'_i))^{i(2^{D-1}-1)}$$

令 $N = \prod_{i=1}^D n_i$ ，这样就得到了一个 $\mathcal{O}(N^2)$ 的算法，不能通过本题。

考虑对某个 $\mathbf{m}$ 的计算，需要考虑其所有长度大于 1 的维，枚举在这些维上如何切分。因此，不妨枚举一些维长度大于 1，另一些维则固定是 1，使用某些方法加速计算出这类巧克力块的答案。这些巧克力块切分成的巧克力块，要么长度大于 1 的维相同，要么这样的维是一个子集。因此按照压位表示从小到大枚举计算即可。

现在考虑如何加速计算，对多项式熟悉的同学很容易看出这是一个多维幂级数的在线卷积。这里不妨还是忽略所枚举的那些固定为 1 的维。那么所有参与计算的下标是

$1 \leq m'_1 \leq m_1, 1 \leq m'_2 \leq m_2, \dots, 1 \leq m'_D \leq m_D$ 。这里，下标从 1 开始不方便多项式计算（事实上，还会影响复杂度，读者可以在读完后文后自行思考），因此这里统一把所有下标减 1，转移式要相应地改成：

$$f(\mathbf{m}) = \sum_{0 \leq m'_1 < m_1 - 1, \dots, 0 \leq m'_D < m_D - 1} f(\mathbf{m}') f(\mathbf{m} - \mathbf{m}' - \mathbf{1}) \times \prod \text{Values of removed chocolates}$$

即需要偏移一个向量 $\mathbf{1}$ 。

接下来依次简单介绍多维幂级数的离线卷积、半在线卷积、在线卷积。离线卷积即参与运算的两个幂级数的值均不依赖卷积的结果，也就是普通的幂级数乘法。对于离线卷积，可将向量 $\mathbf{m}$ 编码成 $\langle n_1, n_2, \dots, n_D \rangle$ 进制数，即 $M = m_1 + m_2 \cdot n_1 + \dots + m_D \cdot n_1 \dots n_{D-1}$ 。对于向量 $\mathbf{m}$ ，定义占位函数 $\chi(\mathbf{m}) = \lfloor \frac{M}{n_1} \rfloor + \lfloor \frac{M}{n_1 n_2} \rfloor + \dots + \lfloor \frac{M}{n_1 n_2 \dots n_{D-1}} \rfloor$ 。将两个下标在 $\langle n_1, n_2, \dots, n_D \rangle$ 进制下相加时，必须不产生进位时才能产生贡献。而读者可以自行思考发现 $\chi(M_1) + \chi(M_2) - \chi(M_1 + M_2)$ 衡量了 $M_1 + M_2$ 时产生进位的次数。显然，这个值取值范围为 $0 \sim D - 1$ 。回顾子集卷积的做法，可以定义某多维幂级数 f 的占位幂级数 $\tilde{f} = \sum_i f_i \mathbf{x}^i t^{\chi(i)}$ 。要求 $f(\mathbf{x})g(\mathbf{x}) \bmod (x_1^{n_1}, \dots, x_D^{n_D})$ 时，即可计算二维幂级数的乘积 $h(x, t) = \tilde{f}(x, t)\tilde{g}(x, t) \bmod (t^D - 1)$ ，显然答案多项式的第 M 项即为 $[x^M t^{\chi(M)}]h(x, t)$ ，其它项因不符合进位规则而不产生贡献。这一卷积的复杂度为 $\mathcal{O}(ND \log N)$ 。

下面考虑多维幂级数的半在线卷积，即参与运算的两个幂级数中，有一个的值依赖卷积结果。注意到 $\langle n_1, n_2, \dots, n_D \rangle$ 进制很自然地赋予了一个半在线卷积的计算顺序，可以使用普通的半在线卷积计算方式，时间复杂度为 $\mathcal{O}(ND \log^2 N)$ 。但是，本题要求比它更优的复杂度（常数小也有可能通过）。考虑每次不是分为左右两个区间，而是分为 k 段区间。注意到每个区间内部 DFT（注意在多维幂级数中是二维 DFT，即两维依次进行 DFT）或 IDFT 只用进行一次，因此每层的复杂度仍为 $\mathcal{O}(ND \log N)$ 。在做卷积时，总共有 $\mathcal{O}(k^2)$ 对区间需要相乘产生贡献，时间复杂度为 $\mathcal{O}(NDk)$ 。显然取 $k = \log N$ 时达到最优复杂度，每层的复杂度仍为 $\mathcal{O}(ND \log N)$ 。但是这样每层可以划分为 $\log N$ 个子区间，递归的层数为 $\log_{\log N} N = \frac{\log N}{\log \log N}$ ，因此总复杂度为 $\mathcal{O}(\frac{ND \log^2 N}{\log \log N})$ 。

在线卷积即参与运算的两个幂级数的值均依赖卷积结果，这也是本题卷积的模式。与半在线卷积同理，多维幂级数的在线卷积与普通幂级数并无差别。假设当前要求 $0 \sim N$ 的在线卷积，令 $N' = \lfloor \frac{N}{2} \rfloor$ ，先求出 $0 \sim N'$ 的在线卷积，再将 $N' + 1 \sim N$ 看做半在线卷积中依赖卷积结果的部分， $0 \sim N'$ 看做半在线卷积中不依赖卷积结果的部分，即可使用半在线卷积算法求解。复杂度显然仍为 $\mathcal{O}(\frac{ND \log^2 N}{\log \log N})$ 。

使用上述算法仍不足以通过本题。注意到需要枚举 2^D 个子集进行计算，因此总复杂度是 $\tilde{\mathcal{O}}((n_1 + 1) \dots (n_D + 1))$ ，无法通过。注意到仍有可以优化的空间。考虑某个向量 $\mathbf{m}$ ，若其某一维为 1，则它只会给后面的向量做贡献。同理，若其某一维为 n_i ，则它只会被更新而不会给别的向量做贡献。如果二者都有，则它不应参与本次枚举的计算。显然，本轮枚举实际参与计算的元素数量是 $\mathcal{O}((n_1 - 1) \dots (n_D - 1))$ 的。这里，不妨先不考虑所有某一维到达上界的元素，先对其它元素进行在线卷积，完成对其它元素的答案的计算。随后，将其它元素进行离线卷积，即可计算出所有某一维达到上界的元素的答案。

总时间复杂度为 $\mathcal{O}(\frac{ND \log^2 N}{\log \log N})$ 。

以上提到的多项式科技，可参见国家集训队 2021 论文集李白天《信息学竞赛中的生成函数计算理论框架》。