# 试题泛做 做题记录

东营市胜利第一中学　姜志豪

# 第一部分　　非 challenge 题目

共 92 题

# 1. Card Shuffle（CARDSHUF）

## 1.1 题目大意

需要设计一个洗牌软件，模拟洗牌的过程。

N 张牌叠成一叠面向下放置在桌子上。从上到下给牌编号，最上面的牌是 1，最下面的牌是 N。重复 M 次下面的步骤：

- 从牌堆顶端拿走 A 张牌。
- 从牌堆顶端拿走 B 张牌。
- 将第一步拿走的 A 张牌按原顺序放回到牌堆顶。
- 从牌堆顶拿走 C 张牌。
- 将第二步拿走的 B 张牌一张张放回到牌堆顶。
- 将第四步拿走的 C 张牌按原顺序放回到牌堆顶。

输出所有操作后的牌堆。

$1 \leqslant N、M \leqslant 10^5$

## 1.2 题目解法

拿走牌，就是删除序列中一部分数。

放回牌，就是在序列中插入一部分数。

一张张放回，就是要将这一部分反转。

这些操作可以用 Splay 来维护：

- 删除前 A 个数，删除的部分记为 a。
- 删除前 B 个数，删除的部分记为 b。
- 将 a 插入到最前面。
- 删除前 C 个数，删除的部分记为 c。
- 将 b 反转，然后插入到最前面。反转可以打标记。
- 将 c 插入到最前面。

最后遍历一遍 Splay 就能求出牌的顺序了。

## 1.3 时空复杂度

时间复杂度：$O(N + M \log N)$

空间复杂度：$O(N)$

# 2. Counting D-sets（CNTDSETS）

## 2.1 题目大意

定义 N 维空间中两个点之间的距离为其各个坐标数值差的最大值。定义一个点集的直径为其中最远的两个点之间的距离。点集中不能有重复的点，点集中的点坐标必须是整数，并且如果两个点集可以相互平移得到，则认为它们是相同的点集。

求 N 维空间中有多少个不同的点集直径为 D。

答案对 10^9+7 取模。

共 T 组数据。

$1 \leqslant T \leqslant 10$，$1 \leqslant N \leqslant 1000$，$1 \leqslant D \leqslant 10^9$

## 2.2 题目解法

直接求直径为 D 的点集数不太好求。可以先求直径不超过 D 的点集数，再减去直径不超过(D-1)的点集数。

对于每一维坐标，点集中所有点这一维坐标的最小值如果必须是 0，那么就不必考虑这一维上的平移问题了。所以，可以加上这个限制，这样就不会重复计算，也不会遗漏。

那么，问题就变成了，求有多少个点集，满足每一维坐标的最小值都是 0，最大值不超过 D。

如果只是限制最小值不小于 0，而不一定必须是 0，这样就很好计算。每一维坐标都可以是[0,D]中的整数，有(D+1)种选择，N 维就有(D+1)^N 种选择。有这么多点可以出现在点集中，所以点集数就是 2^[(D+1)^N]。

可以用容斥原理。先不考虑必须有 0 的限制，直接求方案数。然后减去某一维一定没有 0 点集数，然后加上某两维一定没有 0 点集数……

如果有 t 维必须没有 0，是哪 t 维共有 C(N,t)种情况，所以在计算的时候要加或减去的值是 C(N,t)乘以一种情况的点集的数量。一种情况的点集数的计算就用刚才说过的那种方法。

## 2.3 时空复杂度

时间复杂度：$O(T \cdot N^2)$

空间复杂度：$O(N^2)$

# 3. Sereja and Equality（SEAEQ）

## 3.1 题目大意

佳佳称两个长度为 n 的数组 A,B 相似，如果对于所有 i(1≤i≤n)，满足 C(A,A[i])=C(B,B[i])。其中 C(X,x)等于满足 X[j]<x(1≤j≤n)的 j 的数目。

对于两个排列 P1,P2，佳佳定义函数 F(P1,P2)等于满足 P1[1...r]相似于 P2[1...r](1≤1≤r≤n)并且 P1[1...r]包含不超过 E 个逆序对的数对(1,r)的数目。

现在佳佳对下面这个问题发生了兴趣：对 P1,P2 取遍所有 n 个元素的排列 F(P1,P2)的总和是多少。

共有 T 组数据。

T≤10000，n≤500，E≤$10^6$

## 3.2 题目解法

定义数组 X 的排名数组 X'，表示 X[i]在 X 中是第 X'[i]小的数。

那么，数组 A、B 相似，当且仅当 A'与 B'完全相同。

枚举 P1、P2 中相似的连续子序列 T1 和 T2 的长度 i，对每一个长度 i 单独计算。

T1、T2 的位置可以在[1,i]、[2,i+1]、...、[n-i+1,n]，共(n-i+1)个位置，在每个位置上 T1、T2 相似的次数都是相同的，计算一个位置上的相似次数，乘上(n-i+1)就可以了。考虑计算 P1、P2 在[1,i]位置的子序列 T1、T2 相似的次数。

T1、T2 的排名数组 T'有$i!$种可能。对于一种可能的 T'，在 P1（或 P2）中，这 T1（或 T2）中的 i 个数有$C_n^i$种情况，剩下的(n-i)个数任意排列，有$(n-i)!$种情况，共有$\frac{n!}{i!}$种情况。这$\frac{n!}{i!}$种 P1 和$\frac{n!}{i!}$种 P2 两两配对都能使 T1、T2 相似，所以 T1、T2 会相似$\left(\frac{n!}{i!}\right)^2$次。

设有$f[i][E]$个长度为 i 的 T'满足逆序对数不超过 E。那么，综合以上的分析，长度为 i 的 T1、T2 的相似共出现了$(n-i+1)\cdot f[i][E]\cdot\left(\frac{n!}{i!}\right)^2$次。

所以，最终的答案就是$\sum_{i=1}^{n}(n-i+1)\cdot f[i][E]\cdot\left(\frac{n!}{i!}\right)^2$。

其中 f 数组可以用 DP 求出：

$$f[i][j]=f[i][j-1]+f[i-1][j]-f[i-1][j-i]$$

长度为 i 的 T’的逆序对数最多是 $\frac{i(i-1)}{2}$ 个。那么，如果 E 比 $\frac{n(n-1)}{2}$ 还要大，可以将 E 改为 $\frac{n(n-1)}{2}$，这样不会影响答案。

可以一开始就预处理出 f 数组的值。

## 3.3 时空复杂度

时间复杂度：$O(n^3+Tn)$

空间复杂度：$O(n^3)$

# 4. Sereja and Subsegment Increasings（SEINC）

## 4.1 题目大意

给出两个长度为 n 的数列 A、B，数列中的数都是[0,3]中的整数。一次操作可以将 A 中的一个连续子序列中的数都加 1 然后模 4。求最少多少次操作能把 A 变成 B。

共有 T 组数据。

$1 \leqslant T \leqslant 10$，$1 \leqslant n \leqslant 10^5$

## 4.2 题目解法

设 A[i]被加 1 的次数是 C[i]，那么 $C[i] \equiv B[i] - A[i] (\mathrm{mod}\, 4)$。

假设已经得到了 C[i]，考虑如何计算最少需要的操作次数。可以计算每个位置会作为多少次操作的起点。A[1]一定作为起点 C[1]次。对于每一个 i，满足 $i \geqslant 2$，这 C[i]次操作不必都以 A[i]为起点，可以和 A[i-1]共用操作，所以，以 A[i]为起点的操作数是 $\max(C[i] - C[i-1], 0)$。

那么，问题可以变成这样：有一个数 val，最初是 0，将这个数增加 x 的代价是 x，减小的代价是 0，但 val 不能变成负数。给出 n 次要求，第 i 次的要求是修改 val 使 $val \equiv B[i] - A[i] (\mathrm{mod}\, 4)$。求最小代价。

在达到第 i 次要求的时候，如果 val 减小，只会减小到小于 val 的数中最大的满足同余要求的数，再减小只有可能增大以后的代价，如果 val 增大，只会增大到大于 val 的数中最小的满足同余要求的数，再增大不如以后需要增大时再增大。所以，需要决定的是变小还是变大。

每一步，都先暂时让 val 变小，并且记录一下这时可以用多少代价让 val 增加 4，对应着变大的方案。如果 val 变成负数了，就从之前记录过的能使 val 增加的方案中选择一个代价最小的，让 val 增加。这样求出的就是最小代价了。

## 4.3 时空复杂度

时间复杂度：$O(Tn)$

空间复杂度：$O(n)$

# 5. Sine Partition Function（PARSIN）

## 5.1 题目大意

给出 n、m、x，求：

$$\sum_{k_1+k_2+\ldots+k_m=n}\sin(k_1x)\sin(k_2x)\ldots\sin(k_mx)$$

其中 $k_1,k_2,\ldots,k_m$ 都是非负整数。

共 T 组数据。

1≤T≤10，1≤m≤30，1≤n≤$10^9$，0≤x≤6.28＜2π

## 5.2 题目解法

先考虑 DP 的做法。

定义 $f[n][m]$ 表示 $\sum_{k_1+k_2+\ldots+k_m=n}\sin(k_1x)\sin(k_2x)\ldots\sin(k_mx)$ 的值。

由于三角函数的计算时，sin 和 cos 通常是同时出现的，所以，

再定义 $g[n][m]$ 表示 $\sum_{k_1+k_2+\ldots+k_m=n}\sin(k_1x)\sin(k_2x)\ldots\sin(k_{m-1}x)\cos(k_mx)$ 的值。

转移时分两种情况。一种情况是令原有的 $k_m$ 增加 x，所以从[n-1][m]转移过来。另一种情况是增设一个 $k_m$，此时 $k_m=0$，所以从[n][m-1]转移过来。

根据

$\sin(a+b)=\sin a\cos b+\cos a\sin b$

$\cos(a+b)=\cos a\cos b-\sin a\sin b$

可以得到状态转移方程

$$f[i][j]=f[i-1][j]\cdot\cos x+g[i-1][j]\cdot\sin x+f[i][j-1]\cdot\sin 0$$

$$=f[i-1][j]\cdot\cos x+g[i-1][j]\cdot\sin x$$

$$g[i][j]=g[i-1][j]\cdot\cos x-f[i-1][j]\cdot\sin x+f[i][j-1]\cdot\cos 0$$

$$=g[i-1][j]\cdot\cos x-f[i-1][j]\cdot\sin x+f[i-1][j-1]\cdot\cos x+g[i-1][j-1]\cdot\sin x$$

因为 n 可能会很大，所以要用矩阵乘法来优化转移。

## 5.3 时空复杂度

时间复杂度：$O(Tm^3 \log n)$

空间复杂度：$O(m^2)$

# 6. Dynamic GCD（DGCD）

## 6.1 题目大意

给出一棵有 n 个结点的树，每个点都有一个点权 c，支持两种操作：
1 u v ：查询 u 到 v 路径上所有点的点权的最大公约数
2 u v w ：将 u 到 v 路径上所有点的点权增加 w。
共进行 m 次操作。
1≤n、m≤50000，1≤c≤10000，0≤w≤10000

## 6.2 题目解法

对树进行树链剖分，两种操作就变成序列 A 上的区间加一个数和区间 gcd 的问题了。

因为 $\gcd(a,b)=\gcd(a-b,b)$，那么可以把 A 差分化，令 $B[i]=A[i]-A[i-1]$。

差分化后，修改 A 的一个区间时只会修改 B 中的两个数。查询 A 中区间[l,r]的 gcd 时，相当于求 gcd(A[l],B[l+1],B[l+2],...,B[r])，其中 A[l]对应着 B 的一个前缀和。

那么，就要对 B 序列支持修改一个数、区间求和、区间求 gcd。这可以用线段树来实现。

## 6.3 时空复杂度

时间复杂度：$O(n+m\log^3 n)$

空间复杂度：$O(n)$

# 7. Evil Book（EVILBOOK）

## 7.1 题目大意

Ciel 要收集 666 点魔法力量。

世界上有 n 个厨师，第 i 个厨师有 C[i]点烹饪力量和 M[i]点魔法力量。

Ciel 可以付出 C[i]点努力与第 i 个厨师进行烹饪战斗，获得 M[i]点魔法力量。然后这个厨师的 M[i]值变为 0。

如果 Ciel 拥有至少 m 点魔法力量，就可以付出 m 点魔法力量使一个厨师的 C[i]和 M[i]都变成原来的三分之一。

Ciel 最初没有魔法力量。

求 Ciel 最少付出多少点努力收集至少 666 点魔法力量。

共 T 组数据。

$1 \leqslant T \leqslant 5$，$1 \leqslant n \leqslant 10$，$10 \leqslant m < 666$，$0 \leqslant C[i]、M[i] \leqslant 10^9$

## 7.2 题目解法

对于每一个厨师，枚举改变他的值几次后和他战斗，或不与他战斗。如果和第 i 个厨师战斗时 M[i]≥666，若还能修改他的值，并且修改后仍满足 M[i]≥666，那么一定修改了更好，所以对于第 i 个厨师，M[i]≥666 的情况只有一种，就是不能再修改时的情况。如果和第 i 个厨师战斗时 M[i]＜666，那么情况也只有几种，因为修改太多次数后 M[i]就不比修改付出的魔法力量多了。再加上不与第 i 个厨师战斗的情况，一个厨师的情况最多有 5 种。

那么，直接枚举每个厨师的情况，判断是否可行，取付出的努力最少的方案。

## 7.3 时空复杂度

时间复杂度：$O(T \cdot 5^n)$

空间复杂度：$O(n)$

# 8. Ranka（RANKA）

## 8.1 题目大意

一局围棋可以下很久。

定义题目中的围棋：

考虑在 9×9 的棋盘上进行的棋局。这是一个两人棋类游戏，A 执黑，B 执白，双方交替行棋，A 先手。

如果把相同颜色的棋子按四联通(|x1-x2|+|y1-y2|=1)连接起来，会得到一些联通块。如果一个联通块不与任何一个空点相邻，那么这个联通块就呈“无气”状态。

在每一步中，玩家必须在一个空点放置一个棋子或者选择放弃本轮。如果玩家放置了棋子，可能有下面的情况发生：

•若棋子放置后至少有一个对手的联通块呈无气状态，就将对手呈无气状态的棋子提出盘外，即“提子”。(在这种情况下，我们可以证明在将对手的无气棋子提出盘外后，我方的所有联通块都不会处于无气状态)

•若新放置的棋子会导致至少一个我方的联通块成为无气状态，则不能放置该棋子。

为了避免死循环，有一个规则叫做“禁止全局同形再现”。可以将整个棋局的状态用一个长度为 82 的字符串表示：第一个字符表示当前轮到哪一方，后面的 9×9 个字符表示每一个格点的状态。如果某一步后棋局的状态在之前出现过（字符串相同），那么这一步是不合法的。

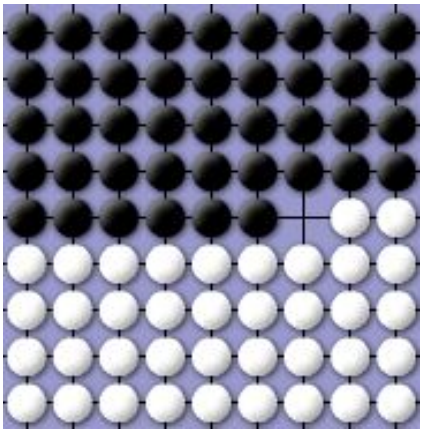
求一种双方落子的方案，使棋局能够进行 n 步。

1≤n≤10000

## 8.2 题目解法

复杂的棋局难以控制局面不重复出现，所以考虑一些简单的落子方案。

棋盘的一边全放黑棋，另一边全放白棋（如右图），改变每次的数量，就可以进行很多步。例如第一次放一个黑棋，剩下位置放白棋，但留出一个位置，然后在那个位置放黑棋，白棋就全部消失了。现在棋盘上有两个黑棋，再在剩下位置放白棋，只留出一个位置，然后在那个位置放黑棋……到最后黑棋几乎占满整个棋盘，这时就超过 5000 步了。



如果本来是黑棋放上面，白棋放下面。再进行一次黑棋放左面，白棋放右面，这样就超过 10000 步了。

在这两种情况之间转换的时候要注意一下，避免出现重复的棋局状态。

## 8.3 时空复杂度

时间复杂度：$O(n)$

空间复杂度：$O(n)$

# 9. Rectangle Query（QRECT）

## 9.1 题目大意

用(a,b,c,d)表示左下角在(a,b),右上角在(c,d)的矩形。

支持三种操作：

- I x1 y1 x2 y2：插入矩形(x1,y1,x2,y2)
- D x：删除第 x 个插入的矩形
- Q x1 y1 x2 y2：求有多少矩形，与(x1,y1,x2,y2)至少有一个公共点

共 m 次操作。

$1 \leqslant m \leqslant 10^5$，$1 \leqslant x1、y1、x2、y2 \leqslant 10^9$

## 9.2 题目解法

先将坐标离散化。

直接求有多少矩形与(x1,y1,x2,y2)有公共点比较难实现，可以用总矩形数减去与(x1,y1,x2,y2)没有公共点的矩形(a,b,c,d)的数量。

- 若 c＜x1，没有公共点
- 若 d＜y1，没有公共点
- 若 a＞x2，没有公共点
- 若 b＞y2，没有公共点

这些数量都可以用线段树来维护。

但有些与(x1,y1,x2,y2)没有公共点的矩形被计算了两次。

- c＜x1，且 d＜y1
- c＜x1，且 b＞y2
- a＞x2，且 d＜y1
- a＞x2，且 b＞y2

维护每一个值都相当于是要支持以下两种操作：

- 加入一个点
- 求一个矩形范围内点的个数

这些操作可以用树状数组套线段树实现，但会超出内存限制。

那么可以用分块的思想：先对每个横坐标建立一棵线段树记录横坐标等于这个值的点的纵坐标有哪些，对横坐标分块，再对每个块建立一棵线段树记录横坐标在这个块范围内的点的纵坐标有哪些。修改就修改横坐标的线段树和对应块的线段树。查询时整块的部分在块的线段树上查询，其他部分在横坐标的线段树上查询。

这样的时间复杂度是$O(m\sqrt{m}\log m)$，但还是会超出时间限制。（大概是我写的常数太大了，有人这样写是过了的。）

既然可以分一次块，那就可以继续分，对已经分出的那些块再进行一次分块，同样对每一个由块组成的大块也建立一棵线段树。修改时要修改大块的线段树、

小块的线段树、横坐标的线段树。查询时先找完整的大块，再在其他部分找完整的小块，剩下的部分就每一个横坐标单独计算。

这样就可以通过所有测试数据了。

## 9.3 时空复杂度

时间复杂度：$O(m\sqrt[3]{m}\log m)$

空间复杂度：$O(m\log m)$

# 10. Music & Lyrics（LYRC）

## 10.1 题目大意

给出 n 个单词（P）和 m 个句子（S）。对于每一个单词，求它在所有句子中共出现了多少次。

单词和句子中只会出现大小写字母、数字和“-”共 63 种字符。

$1 \leqslant n \leqslant 500$，$1 \leqslant |P| \leqslant 1000$，$1 \leqslant m \leqslant 100$，$1 \leqslant |S| \leqslant 50000$

## 10.2 题目解法

对 n 个单词建立 AC 自动机。将 m 个句子在自动机上匹配，每进行一个字符的匹配，就使当前点的被访问次数加一。这样，所有句子匹配完后，一个点的被访问次数就代表这个位置能与句子中的多少个位置匹配。

求出 AC 自动机的 fail 树。对于一个单词，若它在自动机中对应的结点是 A，那么 fail 树中以 A 为根的子树中所有结点被访问次数的和就是这个单词在句子中的匹配次数。从下到上 DP 一遍，就能求出每个子树中的结点被访问次数的和。

## 10.3 时空复杂度

时间复杂度：$O(63n|P|+m|S|)$

空间复杂度：$O(63n|P|)$

# 11. Children Trips（TRIPS）

## 11.1 题目大意

有 n 个旅游区，由 n-1 条道路连接。道路的长度只可能是 1 或 2。

有 m 个学生想要旅行，他会选择起点和终点，然后沿连接它们的唯一路径旅行。同时每个学生都有一个体力 c，表示这个学生一天最多旅行多长的距离。每一天，学生会旅行到能到达的最远的旅游区。

对每个学生，分别求出他们的旅行天数。

1≤n、m≤$10^5$，2≤c≤2n

## 11.2 题目解法

最先能想到的就是模拟每个学生的旅行，从起点开始，求出每一天能旅行到哪个点。查找一天能旅行到哪个点的复杂度是$O(c)$的，天数是$O(\frac{n}{c})$的，所以一次询问的复杂度是$O(n)$的。

学生的旅行方式是从起点 a 开始，每天尽可能多走，走到终点 b。这样需要的时间是体力限制下旅行时间最少的。

任选一个旅游区作为根结点。从 a 向根结点走，每天尽可能多走，直到再走就会到达或经过 a、b 的 lca 才停止，记停止的点为 a’。然后从 b 开始进行同样的过程，记停止的点为 b’。这时，a’、b’之间路需要至多两天完成，lca 两侧都至多再走一天。所以，求出 a’、b’之间的距离，就能判断还需 0 天、1 天还是 2 天。同样，这样需要的时间也是体力限制下旅行时间最少的。

所以，这两种方式求出的时间相同。为了方便，用后一种方式来计算时间。

通过树上倍增的方法，可以将查找一天能旅行到的点优化到$O(\log n)$，那么一次询问就是$O(\frac{n\log n}{c})$的。当 c 较大时，或者说存在一个 t，当$c>t$时，直接这样求就可以了。

当$c\le t$时，这样做效率低的原因是天数很多，所以可以一次走多天。对每一个点 i，预处理出$f[i][j]$表示体力为 j 的学生从 i 出发，向根节点方向走$\left\lfloor\frac{t}{j}\right\rfloor$天，会走到哪个点。询问体力为 c 的学生从 a 到 b 时，先用倍增的方法求出 a、b 的 lca，然后通过 f 数组的值从 a、b 分别向上走，走到不能再连走$\left\lfloor\frac{t}{j}\right\rfloor$天，这时分别到达 a’、b’。然后再用 c 较大时的那种做法求出从 a’到 b’所需时间。这样就求出从 a 到 b 需要的时间了。

## 11.3 时空复杂度

### 11.3.1 时间复杂度

树上倍增预处理：$O(n\log n)$

$f$ 数组预处理：$O(nt\log t)$

$c>t$ 时，一次询问的复杂度是 $O(\frac{n\log n}{t})$。$c\le t$ 时，一次询问的复杂度是 $O(\frac{n}{t}+t\log n)$。所以，一次询问的复杂度是 $O(\frac{n\log n}{t}+t\log n)$。共 m 次询问，询问总复杂度是 $O\left(m\log n(t+\frac{n}{t})\right)$。

所以，这个题目的总复杂度是 $O\left(n(\log n+t\log t)+m\log n(t+\frac{n}{t})\right)$。

当 t 取 $\sqrt{n}$ 时比较优（但不一定是最优），这时，

时间复杂度：$O\left((n+m)\sqrt{n}\log n\right)$

### 11.3.2 空间复杂度

空间复杂度是 $O(n\log n+nt)$。

t 取 $\sqrt{n}$ 时，

空间复杂度：$O\left(n\sqrt{n}\right)$

# 12. The Baking Business（BAKE）

## 12.1 题目大意

帮厨师设计一个智能工具，支持一些操作，以便他的食品公司更有效率。

• I 产品[.大小] 省[.城市[.地区]] 性别 年龄 出售数量

出售操作。包括产品细节，位置细节以及顾客性别和年龄。中括号内的信息表示可能有也可能没有。

• Q 产品[.大小] 省[.城市[.地区]] 性别 起始年龄[-结束年龄]

查询操作。查询在给出的产品、地区、性别、年龄范围内，出售的总数量。中括号内的信息表示可能有也可能没有。若产品为-1，表示查询所有产品。若省为-1，表示查询所有区域。若结束年龄没有，表示只查询起始年龄这一个年龄的信息。

操作次数 n 不超过$10^5$。有 10 种产品，每种都有 3 种不同的大小。有 10 个省份，每个省份可以被划分为 20 个城市，每个城市又可以被划分成 5 个地区。年龄从 1 到 90。

## 12.2 题目解法

直接模拟。

为了提高效率，可以把年龄的存储改为前缀和或树状数组。

信息的读入时可以逐字符读入，边读入边处理。也可以用已有的字符串读入函数读入，然后再对读入的字符串进行处理。

## 12.3 时空复杂度

时间复杂度：$O(n \cdot 90)$

空间复杂度：$O(10 \cdot 3 \cdot 10 \cdot 20 \cdot 5 \cdot 90)$

# 13. Two Companies（TWOCOMP）

## 13.1 题目大意

一个城市的有轨电车即将投入使用，共有 n 个交汇点，由 n-1 条轨道连接起来，形成树形结构。

有两个公司设计了运营线路，一个公司设计了 M1 条，另一个公司设计了 M2 条。一条线路有一个起点、一个终点，还有一个满意度。

两个公司的运营线路不能有相交的部分（没有公共点）。选择一些线路，最大化满意度之和。

$1 \leqslant n \leqslant 10^5$，$1 \leqslant m = \max(M1, M2) \leqslant 700$

## 13.2 题目解法

可以用网络流来求最大满意度之和。一个公司的线路连源点，流量是满意度。另一个公司的线路连汇点，流量是满意度。若一个公司的线路 a 和另一个公司的线路 b 有公共点，在 a、b 之间连一条边，流量是无穷大。所有线路的满意度之和减去最小割就是答案。

通过树上倍增，可以预处理出两个公司的运营线路两两之间是否有公共点。

## 13.3 时空复杂度

时间复杂度：$O(n \log n + m^2 \log n + m^4)$

空间复杂度：$O(n \log n + m^2)$

# 14. Arithmetic Progressions（COUNTARI）

## 14.1 题目大意

给出一个长度为 n 的数列 A，求有多少个三元组(i,j,k)满足 $i<j<k$ 并且 $A[j]-A[i]=A[k]-A[j]$。

$1\leqslant n\leqslant 10^5$，$1\le A[i]\le m=30000$

## 14.2 题目解法

先考虑暴力的做法。

从小到大枚举 j，同时不断更新：$Left[x]$表示在 $A[j]$的前面，x 出现了多少次。$Right[x]$表示在 $A[j]$的后面，x 出现了多少次。枚举 $A[i]$和 $A[k]$的值，借助 $Left[x]$、$Right[x]$的值直接计算。

暴力常数小，这样做就可以通过所有测试数据，甚至比大多数不是暴力的程序运行时间还短。

再考虑如何优化暴力。

可以将原数列分块。同样枚举中间位置 j。分三种情况：

- i、k 都与 j 在同一个块。直接枚举计算就可以。
- i、k 中有一个与 j 在同一个块，另一个与 j 不在同一个块。令 $Left[x]$表示在 $A[j]$所在块的前面，x 出现了多少次。$Right[x]$表示在 $A[j]$所在块的后面，x 出现了多少次。若 i 与 j 在同一个块，就枚举 i，借助 $Right[x]$的值求出合法的 k 的个数。若 k 与 j 在同一个块，就枚举 k，借助 $Left[x]$的值求出合法的 i 的个数。
- i、k 都与 j 不在同一个块。类似暴力的做法，可以枚举 $A[i]$和 $A[k]$的值，借助 $Left[x]$、$Right[x]$的值直接计算。同样，这里的 $Left[x]$表示在 $A[j]$所在块的前面，x 出现了多少次。$Right[x]$表示在 $A[j]$所在块的后面，x 出现了多少次。

可以发现，这是一个卷积的形式，可以用 FFT 优化计算。对于每一个块，都先求出 $Left[x]$、$Right[x]$的值，然后计算卷积。这样，对于每一个块内的 j，包含它

的合法三元组的数量就都计算出来了。

## 14.3　时空复杂度

时间复杂度：$O(n\sqrt{m\log m})$

空间复杂度：$O(n+m)$

# 15. A New Door（ANDOOR）

## 15.1 题目大意

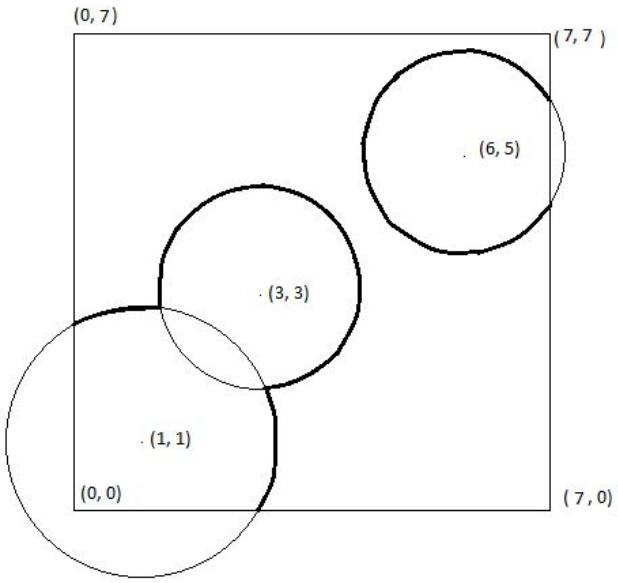
厨师要装饰门，门宽 w 高 h。厨师从门上挖去 n 个圆形，然后在被挖去的部分装上玻璃。一个圆可能只有一部分在门上。圆与圆之间可能相交、包含。

求最终玻璃在门内的边界线总长。例如下图中的粗线部分。

共 T 组数据。

1≤T≤1000，1≤w、h≤1000，1≤n≤1000。

所有数据的 n 的总和不超过 5000。




## 15.2 题目解法

枚举每一个圆 A，计算圆周的哪些部分是边界线。

圆 A 被其他圆覆盖的部分不是边界线，不在门内的部分不是边界线。其他部分都是边界线。那就可以再枚举其他圆 B，计算 A 圆周的哪一部分在 B 内部，记录这一部分不是边界线。再删去 A 不在门内的部分，就求出 A 作为边界线的部分了。

## 15.3 时空复杂度

时间复杂度：$O(n^2 \log n)$

空间复杂度：$O(n)$

# 16. Different Trips（DIFTRIP）

## 16.1 题目大意

有 n 个城市，由 n-1 条道路连接，形成树形结构，其中 1 号点是根结点。

若两个点度数相同，那么它们相似。

厨师要选择路径旅行。厨师先确定一个城市 A，然后在 A 到根的路径上确定一个城市 B，从 A 到 B 就是一条可行的路径。

若两条路径长度相同，且按顺序一一对应的城市都相似，那么这两条路径就是相似的。

求有多少种不相似的可行路径。

$1 \leqslant n \leqslant 10^5$

## 16.2 题目解法

结点的度数相当于一个字符，可行路径上结点的字符组成一个字符串。不相似的可行路径就是字符串不同的可行路径。

如果树是一条链，那么可以用后缀数组来求有多少不同的子串。

在树上，仍然可以用后缀数组来做。对于一个点 A，把它到根结点的路径的字符串看做一个后缀，将这些后缀进行排序，仍然使用倍增的方法。

但是，树上的后缀数组不能直接求出相邻后缀的最长公共前缀，可以二分公共前缀长度，哈希判断。可以通过树上倍增求哈希值。

所有后缀的长度之和，减去排序后相邻后缀的最长公共前缀之和，就是不相似的可行路径数量。

## 16.3 时空复杂度

时间复杂度：$O(n \log n)$

空间复杂度：$O(n \log n)$

# 17. Query on a tree VI（QTREE6）

## 17.1 题目大意

给定一棵 n 个节点的树，每个节点有一个颜色，是黑色或白色，初始都是黑色。

维护一种数据结构，支持下列操作：

- 0 u：询问有多少点到 u 的路径上所有点颜色都相同
- 1 u：改变点 u 的颜色

共 m 次操作。

$1 \leqslant n、m \leqslant 10^5$

## 17.2 题目解法

0 操作中合法的点是联通的，它们当中深度最小的点是唯一的，可以把信息记录在深度最小的点上。

定义 $black[x]$ 表示将 x 的颜色变为黑色，以 x 为根的子树中有多少点到 x 的路径上所有点的颜色都为黑色。定义 $white[x]$ 表示将 x 的颜色变为白色，以 x 为根的子树中有多少点到 x 的路径上所有点的颜色都为白色。

对于 0 操作，先找到合法的点中深度最小的点 x，$black[x]$ 或 $white[x]$ 就是答案。

对于 1 操作，只会影响 u 的部分祖先的 *black* 、 *white* 值。

这些都可以通过树链剖分来维护。

## 17.3 时空复杂度

时间复杂度：$O(n + m\log^2 n)$

空间复杂度：$O(n)$

# 18. Little Elephant and Boxes（LEBOXES）

## 18.1 题目大意

有 n 个盒子，每个盒子里有一些金币或钻石。第 i 个盒子中，有 $p_i$ 的概率有 $v_i$ 个金币，有 $1-p_i$ 的概率有一颗钻石。

有 m 个物品，购买第 i 个物品需要 $c_i$ 个金币和 $d_i$ 颗钻石。一个物品只能买一次。

小象打开所有盒子，获得了其中的金币和钻石。他会购买尽可能多的物品。求他期望能够购买到的物品个数。

共 T 组数据。

$1 \leqslant T \leqslant 5$，$2 \leqslant n \leqslant 30$，$1 \leqslant m \leqslant 30$，$1 \leqslant v_i$、$c_i \leqslant 10^7$

## 18.2 题目解法

通过 dp 预处理出 f[i][j]，表示拥有 i 颗钻石时购买 j 个物品最少需要多少金币。

枚举每个盒子中是金币还是钻石，同时记录当前情况的概率，借助 f 数组的值，得出当前情况最多购买的物品数。这样就能求出期望物品数了。

由于 n 有点大，可以通过折半搜索来优化。

## 18.3 时空复杂度

时间复杂度：$O(2^{\frac{n}{2}}nm+nm^2)$

空间复杂度：$O(2^{\frac{n}{2}}+nm)$

# 19. Black-white Board Game（BWGAME）

## 19.1 题目大意

Alex 和 Fedor 在玩游戏。

他们有一个 n×n 的网格，初始所有格子都是白色的。Alex 将第 i 行中第 l[i]列到第 r[i]列的格子涂黑。

每一轮游戏中，两个人各找出一个 1~n 的排列。Alex 的排列逆序对数必须是偶数，Fedor 的排列逆序对数必须是奇数。之前找出过的排列不能再用。

若两人都找出，进行下一轮。若一人找出一人没有找出，找出的人获胜。若两人都没有找出，游戏平局。

Alex 和 Fedor 很聪明，有就一定能找出。求最后谁胜利，或是平局。

共进行 T 次游戏。

- 子任务 1：1≤T≤1000，1≤n≤7
- 子任务 2：1≤T≤100，1≤n≤100
- 子任务 3：1≤T≤15，1≤n≤$10^5$

## 19.2 题目解法

题目实际上就是：

一个 1~n 的排列合法，当且仅当，排列中第 i 个数在 l[i]和 r[i]之间。求合法排列中，逆序对数是奇数的排列多，还是逆序对数是偶数的排列多，还是一样多。

交换一个排列中任意两个数，排列的逆序对数的奇偶性发生改变。

对于排列 P，P[i]在区间[a,b]中选择，同时 P[j]也在区间[a,b]中选择，那么逆序对数是奇数的 P 和逆序对数是偶数的 P 一样多。这是因为任意一个合法的 P，交换 P[i]和 P[j]的值，仍然合法，但奇偶性改变。

假设 a、b 满足 l[i]≤a≤b≤r[i]，l[j]≤a≤b≤r[j]。若禁止 P[i]和 P[j]同时在区间[a,b]中的情况，对最终的大小比较没有影响，因为被禁止的奇数的情况和偶数的情况同样多。

从小到大考虑排列中的数。先考虑哪个位置等于 1，再考虑 2、3 直到 n。

先考虑哪个位置等于 1。若不存在 i 满足 l[i]=1，那么就没有合法的 P，游戏平局。若只有一个 i 满足 l[i]=1，那么必定是 P[i]=1。若存在多个 i 的满足 P[i]=1，它们分别是 i[1]、i[2]、...、i[k]，不妨令 r[i[1]]≤r[i[2]]≤...≤r[i[k]]。根据之前的结论，可以禁止 P[i[2]]、P[i[3]]、...、P[i[k]]与 P[i[1]]同时在区间[1,r[i[1]]]中取值。P[i[1]]必定在[1,r[i[1]]]内取值，所以禁止 P[i[2]]、P[i[3]]、...、P[i[k]]在[1,r[i[1]]]内取值。那么就可以将 l[i[2]]、l[i[3]]、...、l[i[k]]的值修改为 r[i[1]]+1。现在，只有一个 i 满足 l[i]=1，那就是 i[1]。那么，1 的位置就确定了。

用同样的方法，除了中途判断游戏平局的情况，n 个数的位置都可以确定下来，能够得到一个排列。求出这个排列逆序对数的奇偶性，就得到了游戏的结果。

实现这个过程时，可以对每一个区间左端点的值 t 都建立一个堆，来存储 l[i]的值为 t 的所有位置 i，在堆中以 r[i]为关键字排序，这样能求出 i[1]的值。l[i[2]]、l[i[3]]、...、l[i[k]]的修改，就是两个堆的合并。合并时将结点数少的堆中的结点逐一加入结点数多的堆中，对于一个结点，它以单点的形式加入到一个更大的堆中的次数是 $O(\log n)$ 的，因为每一次加入到一个更大的堆中，它所在的堆的大小都会至少是原来的两倍。

## 19.3 时空复杂度

时间复杂度：$O(Tn\log^2 n)$

空间复杂度：$O(n)$

# 20. Observing the Tree（QUERY）

## 20.1 题目大意

给出一棵树，有 n 个点。点有点权，初始都是 0。支持三种操作：

• 1 x y a b：在 x 到 y 的路径上，第一个点（x）的点权加上 a，第二个点的点权加上 a+b，第三个点的点权加上 a+2b……

• 2 x y：询问 x 到 y 的路径上所有点的点权和。

• 3 x：将所有点的点权都改为第 x 次 1 操作后的值。

共 m 次操作，强制在线。

$1 \leqslant n、m \leqslant 10^5$

## 20.2 题目解法

通过树链剖分，可以把所有操作变成链上的问题。下面只考虑在链上的实现方式。

对于修改操作，若将原序列 $A$ 差分化成 $B$，那么在 $B$ 中，除区间端点附近位置外，区间内部的值都要增加 b，区间外部的值不变。再差分化一次，将 $B$ 差分化成 $C$，那么在 C 中，除区间端点附近外，其他位置的值都不变。所以通过两次差分化，能够将修改操作变成单点修改。

对于查询操作，假设求[1,r]的和，可以先求出[1,r]的和，再减去[1,l-1]的和。

因为 $B[x]=\sum_{i=1}^{x} C[i]$，$A[x]=\sum_{i=1}^{x} B[i]$，

所以 $A[x]=\sum_{i=1}^{x}(x+1-i)\cdot C[i]$，

那么，

$$\begin{aligned}
&\sum_{j=1}^{x} A[j] \\
&=\sum_{i=1}^{x}\sum_{j=i}^{x}(j+1-i)\cdot C[i] \\
&=\sum_{i=1}^{x}\frac{(x+1-i)(x+2-i)}{2}\cdot C[i] \\
&=\sum_{i=1}^{x}\frac{i^2}{2}\cdot C[i]-\sum_{i=1}^{x}\frac{(2x+3)\cdot i}{2}\cdot C[i]+\sum_{i=1}^{x}\frac{(x+1)\cdot(x+2)}{2}\cdot C[i]
\end{aligned}$$

所以，对 $i^2\cdot C[i]$、$i\cdot C[i]$、$C[i]$ 分别求和，就可以求出 $A$ 的前缀和了。可以

通过线段树来实现。

对于回到之前某时刻的状态的操作，可以用可持久化线段树来实现。

## 20.3 时空复杂度

时间复杂度：$O(n+m\log^2 n)$

空间复杂度：$O(m\log^2 n)$

# 21. Gangsters of Treeland（MONOPLOY）

## 21.1 题目大意

树之大陆是一个有 n 座城市的王国，其中有一个城市是首都。城市之间有 n-1 条双向道路连接，形成树形结构，首都是根节点。

最初，每个城市都被不同的帮会控制。居民可以沿道路从一个城市到另一个城市，若一条道路连接的两个城市属于不同的帮会，那么经过这条道路需要付出一个单位的代价，若属于同一帮会，那么不需要付出代价。

会有两种事件发生：

- 1 x：出现了一个新的帮会，控制了从 x 到首都路径上所有城市。
- 2 x：求以 x 为根的子树中的城市到首都的代价的平均值。

共发生 m 次事件。

共 T 组数据。

$1 \leqslant T \leqslant 15$，$1 \leqslant n、m \leqslant 10^5$。

所有数据 n 的总和不超过 $2 \times 10^5$。

所有数据 m 的总和不超过 $2 \times 10^5$。

## 21.2 题目解法

新的帮会的出现类似于 LCT 中的 Access 操作。初始所有边都是轻边，每次出现新的帮会，就对那个点进行 Access 操作，将那个点到根节点的路径上所有边都改为重边。轻边的代价就是 1，重边的代价就是 0。

求子树中的城市到首都的代价的平均值，可以求出子树中每个点到首都的代价的和，再除以点的数量。每次一条边的代价发生改变时，就修改影响到的点到根结点的代价。需要修改的是一个子树中的所有点，可以求出每个点的 dfs 序，子树中的点 dfs 序是连续的，可以用线段树来维护。

## 21.3 时空复杂度

时间复杂度：$O(n + m\log^2 n)$

空间复杂度：$O(n)$

# 22. Two k-Convex Polygons（TKCONVEX）

## 22.1 题目大意

给出 n 根木棒的长度，从中选出 2m 根木棒，组成两个凸 m 边形。

判断是否存在方案，若存在，输入任意一种。

1≤n≤1000，3≤k≤10

## 22.2 题目解法

可以发现，若长度从小到大分别为 a[1]、a[2]、...、a[m]的木棒能够组成凸 m 边形，当且仅当 a[1]+a[2]+...+a[m-1]>a[m]。

先将木棒按长度排序。

若只需组成一个 m 边形，可以枚举 a[m]，剩下 m-1 根木棒就选择长度不超过 a[m]的木棒中最大的 m-1 根，判断能否组成凸 m 边形。

若通过这种寻找方案的方式找到的合法方案数小于 2，那么就无解。

若找出至少两种组成 m 边形的方案没有使用相同的木棒，那么这就是一组解。

现在只剩下一种情况，就是存在两种以上组成 m 边形的方案，任意两种方案都有共用的木棒。那么方案数不超过 k。枚举两个 m 边形中的 a[m]分别这 $O(k)$ 种方案中的哪一种，剩下的 2m-2 根木棒尽可能取长的，能够确定下来选哪 2m-2 根。枚举这些木棒各属于哪个多边形，判断是否合法。

## 22.3 时空复杂度

时间复杂度：$O(n\log n+k^2C_{2k}^k)$

空间复杂度：$O(n)$

# 23. Counting The Important Pairs（TAPAIR）

## 23.1 题目大意

给出一张 n 个点 m 条边的无向图。删去两条边，使图不连通，求方案数。

$1 \leqslant n \leqslant 10^5$，$1 \leqslant m \leqslant 3 \times 10^5$

## 23.2 题目解法

若原图不连通，那么所有删边方案都合法。下面考虑图连通的情况。
对图进行 dfs，建立 dfs 树。
删除两条非树边图仍然连通。
删去一条树边一条非树边，若图不连通，有两种情况：
- 树边不被任何非树边覆盖。
- 树边仅被删去的那条非树边覆盖。

删去两条树边，若图不连通，有两种情况：
- 至少一条删去的树边不被任何非树边覆盖。
- 覆盖这两条树边的非树边都同时覆盖了这两条树边。

可以通过哈希的方法求出方案数。
给每一条非树边随机赋值，树边的值是所有覆盖它的非树边的值的异或和。
若一种删边方案使图不连通，当且仅当满足下列两个条件中的至少一个：
- 删去了一条值为 0 的边。
- 删去的两条边值相等。

对所有边的值排序，就可以求出满足条件的方案数。

## 23.3 时空复杂度

时间复杂度：$O(n + m\log m)$

空间复杂度：$O(n + m)$

# 24. Chef and Churu（FNCS）

## 24.1 题目大意

有一个长度为 n 的数组 A，元素编号 1~n。还有 n 个函数，函数编号 1~n，第 i 个函数的返回值是 A 中编号在 L[i]和 R[i]之间的数的和。

支持两种操作：

- 1 x y：将 A[x]修改为 y。
- 2 l r：求编号在 l 和 r 之间的函数的返回值之和。

共 m 次操作。

$1 \leqslant n、m \leqslant 10^5$

## 24.2 题目解法

一个函数的返回值可以通过线段树求出。

对函数按编号进行分块。维护每一块内的函数的返回值之和。

预处理出每一块内的函数统计 A 中的每个元素各多少次。

修改 A 中的值时，根据每个块中统计该数的次数，修改每一个块的和。同时修改线段树的值。

查询时，整块的部分的和已被计算，其他部分通过线段树逐个函数计算。

## 24.3 时空复杂度

时间复杂度：$O(n\sqrt{n}+m\sqrt{n}\log n)$

空间复杂度：$O(n\sqrt{n})$

# 25. Prime Distance On Tree（PRIMEDST）

## 25.1 题目大意

给出一棵 n 个点的树，树边的长度都是 1。从中随机取两个不同的点，求这两个点的距离是质数的概率。

2≤n≤50000

## 25.2 题目解法

距离是质数的点对数，除以选择两个不同的点的方案数，就是两个点距离是质数的概率。

选择两个不同的点的方案数就是$C_n^2$。

距离是质数的点对数，可以通过树分治求出。

分治到一部分时，如果这一步要删除的点是 A 点，需要计算路径经过 A 点且距离为质数的点对数。直接计算点对数需要枚举点对中的两个点，效率较低。可以发现枚举的计算实际上是计算了一个卷积，所以可以通过 FFT 来优化。

## 25.3 时空复杂度

时间复杂度：$O(n\log^2 n)$

空间复杂度：$O(n)$

# 26. Xor Queries（XRQRS）

## 26.1 题目大意

给定一个初始为空的整数序列 A，支持下列操作：

- 0 x：在数组最后加入数字 x。
- 1 l r x：在区间 A[l...r]中，找一个数字 y，最大化(x xor y)。
- 2 x：删除数组最后 x 个数。
- 3 l r x：在区间 A[l...r]中，统计小于等于 x 的数的个数。
- 4 l r x：在区间 A[l...r]中，找到第 x 小的数。

数组中的数是不超过 t 的正整数。

共 m 次操作。

$1 \leqslant m \leqslant 5 \times 10^5$，$1 \leqslant t \leqslant 5 \times 10^5$

## 26.2 题目解法

可以把数看做 01 串，用可持久化字典树来维护。

操作 0、2 就是在可持久化字典树中的增加和删除。

操作 1 中，可以从高位向低位枚举，若 y 能与 x 这一位不同，就使它们不同，否则就相同。

操作 3 中，可以改成统计小于(x+1)的数的个数。从高位向低位枚举，若(x+1)的这一位是 1，就把之前位相同，这一位不同的数的个数加入答案中。

操作 4 中，可以从高位向低位枚举，设这一位为 0 的数有 y 个。若 $x \leqslant y$，那么第 x 小的数这一位是 0。否则第 x 小的数这一位是 1，并将 x 减去 y。

## 26.3 时空复杂度

时间复杂度：$O(m \log t)$

空间复杂度：$O(m \log t)$

# 27. Course Selection（RIN）

## 27.1 题目大意

铃现在正在大学学习。

课业计划共包含 n 项课程，每项课程都需要在 m 个学期里的某一个完成。

一些课程有前置课程，a 是 b 的前置课程表示课程 a 的完成的学期要在课程 b 完成的学期的前面。共有 k 组课程顺序要求。

对于课程 i，在不同学期完成的得分不同。令 x[i][j]表示课程 i 在学期 j 完成的得分。若 x[i][j]为-1，表示学期 j 中没有开设课程 i。

计算铃各个课程分数的平均值的最大值。

1≤n、m≤100，0≤k≤100，-1≤x[i][j]≤100

保证存在课程安排方案。

## 27.2 题目解法

各课程分数的平均值，就是各课程分数之和除以课程数。所以就是要最大化课程分数之和。得分最大就是扣分最小，可以假设满分是 100 分，最小化扣分之和。可以考虑通过最小割来解。

令点(i,j)表示课程 i 在学期 j 学习。从源点 S 连向(i,1)一条边，容量为 x[i][1]。对于 j>1，从(i,j-1)连向(i,j)一条边，容量为 x[i][j]。从(i,m)连向汇点 T 一条边，容量为 inf。这样求出的最小割，就是不考虑前置课程要求的情况下，最小扣分。

若 a 是 b 的前置课程，那么 a 的割边位置要在 b 的割边位置的前面。可以从 S 向(b,1)连一条容量为 inf 的边。对于 j>1，从(a,j-1)向(b,j)连一条容量为 inf 的边。这样就能强制 a 的割边位置在 b 的割边位置的前面。

## 27.3 时空复杂度

时间复杂度：$O\left(n^2m^2\left(nm+km\right)\right)$

空间复杂度：$O\left(nm+km\right)$

# 28. Shortest Circuit Evaluation（SHORTCIR）

## 28.1 题目大意

布尔型的短路计算是指如果第一个变量的值已经足够得到结果，第二个布尔型变量的值就不会被计算。

给出一个长度为 n 的布尔型表达式，共包含 m 个变量，给出每个变量值为 true 的概率。除变量外，仅包含 and、or、not 和括号。安排计算变量的顺序，按该顺序计算变量并且不会计算那些在过程中不可能影响表达式值的变量。

求变量计算个数的期望值的最小值。

共 T 组数据。

1≤T≤50，1≤n≤30000，1≤m≤1000

## 28.2 题目解法

运算符 not 可以直接算进变量里，变量原来值为 true 的概率是 a，算进 not 后就变成 1-a。下面不考虑 not。

由于括号的存在和 and、or 优先级不同，所以大多数运算顺序已经确定，能任意改变运算顺序的只有形如 a and b and c...和 a or b or c...的情况。

考虑 and 运算。假设 a、b 在表达式中相邻。若 a 的期望计算次数为 x[a]，值为 true 的概率为 y[a]，b 同样有 x[b]、y[b]。若运算顺序为 a and b，a 必定被计算，b 有 y[a]的概率被计算，a and b 的期望运算次数为 $x[a]+y[a]\cdot x[b]$。

若运算顺序为 b and a，那么期望运算次数为 $x[b]+y[b]\cdot x[a]$。根据这两个值，就能确定任意两个相邻变量是否需要交换运算顺序，那么所有变量的运算顺序就都能通过一次排序确定。

运算 or 的处理方式与运算 and 类似。

## 28.3 时空复杂度

时间复杂度：$O(T(n+m\log m))$

空间复杂度：$O(n)$

# 29. Devu and Locks（DEVLOCK）

## 29.1 题目大意

求有多少 n 位十进制数是 p 的倍数且每位之和小于等于 t，允许前导 0。

对每一个 $t \in [0,m]$ 输出一个答案。

答案对 998244353 取模。

- $1 \leqslant n \leqslant 10^9$，$1 \leqslant p \leqslant 50$，$1 \leqslant m \leqslant 500$
- $1 \leqslant n \leqslant 10^9$，$1 \leqslant p \leqslant 16$，$1 \leqslant m \leqslant 15000$

## 29.2 题目解法

定义 $f[i][j][k]$ 表示有多少 i 位数每位之和为 k，且除以 p 余 j。

那么，

$f[i][j][k] = \sum f[i-1][a][b] \cdot f[1][c][d], (a+c) \bmod p = j, b+d = k$

由于 n 很大，考虑类似于倍增的 dp 方式，

$f[2i][j][k] = \sum f[i][a][b] \cdot f[i][c][d], (a+c) \bmod p = j, b+d = k$

所以，可以通过类似于快速幂的方式计算 $O(\log n)$ 个 $f[i]$ 来得到 $f[n]$。

在通过 $f[x]$ 和 $f[y]$ 的值计算 $f[x+y]$ 时，可以通过 NTT 来优化。

## 29.3 时空复杂度

时间复杂度：$O(pm \log m \log n + p^2 m \log n)$

空间复杂度：$O(pm)$

# 30. Count on a Treap（COT5）

## 30.1 题目大意

给出一个空的大根堆 Treap。按照关键字，它是二叉查找树。按照权值，它是大根堆。支持下列操作：

- 0 x y：插入一个关键字为 x，权值为 y 的点。
- 1 x：删除关键字为 x 的点。
- 2 x y：返回关键字分别为 x 和 y 的两个结点的距离。

共 n 次操作。

$1 \leqslant n \leqslant 2 \times 10^5$

## 30.2 题目解法

查询操作时，若将 Treap 中的结点按关键字排序得到结点序列 A，那么 x、y 在 Treap 中的 lca 就是 A 中以 x、y 为端点的区间中权值最大的点。又因为，

$$x、y\text{的距离} = x\text{的深度} + y\text{的深度} - 2 \cdot lca\text{的深度}$$

那么，通过求点的深度就可以求出两点的距离了。

深度就是祖先数+1。若 y 是 x 的祖先，那么在 A 中以 x、y 为端点的区间中权值最大的点就是 y。

离线，将所有点的关键字离散化，就能通过线段树来支持插入、删除、查找祖先数了。

## 30.3 时空复杂度

时间复杂度：$O(n \log^2 n)$

空间复杂度：$O(n)$

# 31. Count Special Matrices（SPMATRIX）

## 31.1 题目大意

$A[1..n][1..n]$是一个 n×n 的矩阵。

满足下列条件的矩阵被称为特殊的：

- $A[x][x]=0,1\le x\le n$
- $A[x][y]=A[y][x]>0,1\le x<y\le n$
- $A[x][y]\le \max(A[x][z],A[z][y]),1\le x,y,z\le n$
- $A[x][y]\in[1,n-2],1\le x<y\le n$
- 对于任意的$k\in[1,n-2]$，存在 x、y 满足$A[x][y]=k$

求特殊矩阵的数量。
共 T 组数据。

$1\leqslant T\leqslant 10^5$，$3\leqslant n\leqslant 10^7$

## 31.2 题目解法

可以得到 n×n 的特殊矩阵的数量是，

$$\frac{n!(n-1)!}{3\cdot 2^{n-1}}\left(\frac{3n}{2}-2-\sum_{i=1}^{n-1}\frac{1}{i}\right)$$

可以预处理出数据范围内所有 n 的答案。

因为要用到逆元，可以预处理出数据范围内所有数的逆元。有一种线性时间复杂度的预处理方法：

假设在模 p 的意义下，求 i 的逆元。

存在 k、r 满足$k\cdot i+r=p,0\le r<i$，其中$k=\left\lfloor\frac{p}{i}\right\rfloor$，$r=p \bmod i$。

那么，

$k\cdot i+r\equiv 0(\bmod p)$

$k\cdot r^{-1}+i^{-1}\equiv 0(\bmod p)$

$i^{-1}\equiv -k\cdot r^{-1}(\bmod p)$

根据这个式子就能线性预处理逆元了。

## 31.3　时空复杂度

时间复杂度：$O(n)$

空间复杂度：$O(n)$

# 32. Annual Parade（PARADE）

## 32.1 题目大意

魔法大陆上有 n 座城市，城市之间有 m 条有向道路相连，每条道路都有经过一次所需要的费用。

每一年英雄们都会进行游行，第 i 个英雄的游行从城市 begin[i]开始，途经一些城市，最终在城市 end[i]结束。路径上至少有两个不同的城市。

游行费用有三部分：

·沿道路移动的总费用。

·若 $begin[i] \neq end[i]$，需要 c 的费用送这位英雄回家。

·如果某个城市没有被任何一位英雄经过，需要 c 的费用来赔偿该市市民。

c 的值每年都可能会改变，已经知道了接下来 k 年每一年的 c 值，求每一年的最小花费。

2≤n≤250，1≤m≤30000，1≤k≤10000

## 32.2 题目解法

用 Floyd 算法预处理出城市两两之间的最短距离。

可以使一个城市只被一位英雄经过。若城市 A 已经被一位英雄经过了，如果其他英雄要从 B 路过 A 到 C，那么可以看做直接从 B 到 C，距离已经预处理出来了。

这样，一个城市 A 可能有三种情况：

·不被任何英雄经过，需要费用 c。

·有英雄从另一个城市 B 过来，费用是 B 到 A 的最短距离。

·有英雄以这里为起点，费用是 c。

有一种特殊的情况，就是英雄从 A 出发又回到 A，这样虽然以 A 作为起点，但回家不需要费用，所以不必计算费用 c。这时，A 的情况会被作为第二种情况，没有计算费用 c。

第一、三种情况费用都是 c，可以合并起来，就是没有另一个城市 B 到达 A。

第二种情况就是一个城市 B 到达 A，费用是 B 到 A 的最短距离。B 的下一个点是 A，就不能再出现 B 的下一个点是其他城市 C 的情况了。这就是二分图的匹配问题了。

S 集中 n 个点代表 n 个城市，T 集中 n 个点代表 n 个城市。S 集中的点 i 与 T 集中的点 j 连边，边权是 i、j 的距离，这里要求 $i \neq j$。那么，

$$\text{方案的费用} = \text{匹配边的边权和} + \mathrm{c} \cdot (n - \text{匹配边的数量})$$

对于每一个匹配边数量 i，都求出一个值 val[i]表示匹配边数量为 i 时，匹配边的边权和最小是多少。这可以通过最小费用流来求。

对于每一次询问，枚举匹配边的数量，求出方案费用的最小值。

## 32.3 时空复杂度

时间复杂度：$O(n^3+nk)$

空间复杂度：$O(n^2)$

# 33. Divide or die（DIVIDEN）

## 33.1 题目大意

有一个 n 度角，n 是整数。给出角的顶点和两条边上各一点。通过尺规作图，将角 n 等分。

尺规作图允许的操作：

- Line x y：作过 x、y 的直线
- Circle x y z：以 x 为圆心，y、z 的距离为半径，作圆。

操作中的 x、y、z 必须是已知的。

输入的三个点是已知的，作图时作出图形的交点也是已知的。

$0<n<360$

## 33.2 题目解法

若 n 是 3 的倍数，无解。

否则，作一个正五边形，内角是 108 度。作一个正三角形，内角是 60 度。这样可以得到 48 度角。不断作角平分线，能够得到 3 度角。用 n 度角不断减 3 度角，最终得到 1 度或 2 度角。若得到 2 度角，再作角平分线。这样得到了一个 1 度角。

通过一个 1 度角就能将 n 度角 n 等分了。

## 33.3 时空复杂度

时间复杂度：$O(n)$

空间复杂度：$O(1)$

# 34. Find a special connected block（CONNECT）

## 34.1 题目大意

给出一个 n×m 的矩阵，每个格子上填着$[-1,n\cdot m]$中的一个整数。

你的任务是找到一个连通块(只能通过上下左右四个方向连接)，要求在这个连通块里，至少要包含 k 个不同的正数，且不能有-1。

给出选取每一个格子所需要的代价，最小化选出的连通块所包含的格子的代价和。

1≤n、m≤15，1≤k≤7

## 34.2 题目解法

若格子中的数都在都不超过 k，那么这就是最小斯坦纳树的问题了。可以通过 dp 解决。

可以将$[1,n\cdot m]$中的数映射到$[1,k]$上，然后用最小斯坦纳树的方法求出最优值。多次随机映射方式，取最小值。

映射后的最优值是原问题最优值的概率是$\frac{k!}{k^k}$。

## 34.3 时空复杂度

若进行 t 次随机，

时间复杂度：$O(3^k tnm)$

空间复杂度：$O(2^k nm)$

# 35. Find a Subsequence（FINDSEQ）

## 35.1 题目大意

给出一个长度为 n 的数组 A[0],A[1],...,A[n-1]和一个字符串 B，B 是“12345”的排列。需要找到一个长度为 5 的 A 的子序列，该子序列中的元素互不相等，并满足他们的相对大小和 B 一样。

也就是说，如果(i0,i1,i2,i3,i4)是找到的 A 的子序列的下标，并且其中

0≤i0<i1<i2<i3<i4<n

那么，

A[i0]是五个数中第 B[0]小的，

A[i1]是五个数中第 B[1]小的，

A[i2]是五个数中第 B[2]小的，

A[i3]是五个数中第 B[3]小的，

A[i4]是五个数中第 B[4]小的。

输出任一方案。

共 T 组数据。

T≤20，5≤n≤1000

## 35.2 题目解法

枚举 i1、i3，枚举时要满足 A[i1]、A[i3]之间的大小关系。为了满足 A[i0]、A[i2]和 A[i4]这三个数与 A[i1]和 A[i3]这两个数之间的大小关系，A[i0]、A[i2]、A[i4]各有一个取值范围。那么就只需要再考虑 A[i0]、A[i2]、A[i4]之间的大小关系了。

A[i0]、A[i2]、A[i4]中，最大化最大的数，最小化最小的数，第二小的数的取值范围就确定的。查找是否存在一个数能够作为第二小的数。

位置 i1 和 i3 通过枚举得到，位置 i0、i2 和 i4 可以通过线段树查找到。

## 35.3 时空复杂度

时间复杂度：$O(Tn^2 \log n)$

空间复杂度：$O(n)$

# 36. Chef and Graph Queries（GERALD07）

## 36.1 题目大意

有一个无向图 G。顶点从 1 到 n 标号，边从 1 到 m 标号。

有 q 对询问 Li，Ri。对于每对询问，求出当仅保留编号在[Li,Ri]中的边时，图 G 中有多少连通块。

共 T 组数据。

1≤T≤1000

所有 n 的和不超过 $2\times10^5$

所有 m 的和不超过 $2\times10^5$

所有 q 的和不超过 $2\times10^5$

## 36.2 题目解法

如果每次询问将边逐一加入图中，可以用并查集维护连通性。

对边按编号进行分块。

对于区间左右端点在同一块内的询问，可以这样来做。

对于区间左右端点不在同一块内的询问，可以用类似于莫队的方法求出每次询问的结果。

用莫队算法的排序方式将区间排序。左端点在同一块内的区间同时处理。假设当前枚举到的左端点所在块为 A，A 中编号最大的边是 i。那么从 i 到 m 依次加边，即加边顺序是 i、i+1、i+2、……、m。对于区间[l,r]，若已经将 r 加入并查集，那么还需要加入[l,i)，它们都在 A 中。这时，用类似 dfs 的方法将这些边逐一加入，求出这个区间的答案，然后在 dfs 结束时还原对并查集的修改。

## 36.3 时空复杂度

时间复杂度：$O(n+m\sqrt{q})$

空间复杂度：$O(n+m+q)$

# 37. The Street（STREETTA）

## 37.1 题目大意

大街上有 n 个商店，依次编号 1 到 n。顾客在某个商店购买一件纪念品的花费为纪念品的花费加上税费。初始时所有商店均没有纪念品，且税费为 0。

维护以下操作：

1 u v a b：新增纪念品操作，对于编号在[u,v]中的商店，编号为 u+i 的商店，新增一种价格为 b+ia 的商品。

2 u v a b：税费调整操作，对于编号在[u,v]中的商店，编号为 u+i 的商店，税费增加 b+ia。

3 i：询问操作，询问顾客在编号为 i 的商店，购买一件纪念品最高的花费是多少。

共 m 次操作。

$1 \leqslant n \leqslant 10^9$

$1 \leqslant m \leqslant 3 \times 10^5$

## 37.2 题目解法

离线，将纪念品商店编号离散化，只维护会被询问到的纪念品商店的信息。

对于纪念品和税费分别维护。

税费可以通过在线段树上打标记维护。

纪念品最大价格也可以用线段树来维护，但标记不便于合并。

对于线段树中的一个点 A，它的两个子结点是 L、R。假设需要在 A 打标记 f，原有标记为 g。由于 f、g 都是关于商店编号 x 的一次函数，当取较大值时，取 f 的值的 x 范围和取 y 的值的 x 范围都是连续的。所以，f、g 中存在一个函数的作用区间不会同在 L、R 中，将这个函数标记下传，另一个标记留在 A 处。

## 37.3 时空复杂度

时间复杂度：$O(m \log^2 m)$

空间复杂度：$O(m)$

# 38. Queries on tree again!（QTREE）

## 38.1 题目大意

有一个包含 n 个节点和 n 条边的简单无向连通图，图中仅包含一个环，保证环的长度为奇数。

节点是从 1 到 n 标号的。每条边对应着一个整数权值。

定义节点 u 到 v 的最短路径为连接 u、v 的边数最少的路径。

模拟以下两种操作：

- f u v：对 u 到 v 的最短路径上的所有边的权值取相反数。
- ? u v：在 u 到 v 的最短路径上，找一段连续的子路径，使得子路径中边的权值之和最大。

共 m 次操作。

$1 \leqslant n \leqslant 10^5$，$1 \leqslant m \leqslant 10^5$

## 38.2 题目解法

如果是在一条链上，可以用线段树维护。

对于线段树中的每一个点，需要记录对应区间中的信息：

- 从区间左端选取至少一个数，最小、最大是多少
- 从区间右端选取至少一个数，最小、最大是多少
- 从区间中选取一段连续的数，最小、最大是多少
- 区间中所有数的和是多少

通过这些信息，就可以对区间中的数进行取反，也可以合并两个区间的信息。

如果在树上，可以对树进行树链剖分。

如果有一个环，可以单独维护环上的信息，环上每一个点，都可以看做是一棵树的根结点，那么所有不在环上的点，都属于一棵以环上一点为根的树。树的部分可以树链剖分，环的部分可以将环断成一条链维护。

## 38.3 时空复杂度

时间复杂度：$O(n + m\log^2 n)$

空间复杂度：$O(n)$

# 39. Short II（SHORT2）

## 39.1 题目大意

给出一个质数 p，求有多少对 a，b（a>p，b>p）满足 ab 被(a-p)(b-p)整除。共 T 组数据。

$1 \leqslant T \leqslant 5$，$1 \leqslant p \leqslant 10^{12}$

## 39.2 题目解法

$(a-p)(b-p) \mid ab, a > p, b > p$

用 $a$ 代替 $a-p$，用 $b$ 代替 $b-p$，可以得到，

$ab \mid (a+p)(b+p), a > 0, b > 0$

那么，

$ab \mid ab + ap + bp + p^2$

$ab \mid ap + bp + p^2$

分三种情况考虑，

• ① $a \bmod p = 0, b \bmod p = 0$

用 $ap$ 代替 $a$，用 $bp$ 代替 $b$，可以得到，

$abp^2 \mid ap^2 + bp^2 + p^2$

$ab \mid a + b + 1$

可以发现，只有 5 对 a，b 满足条件。

• ② $a \bmod p \neq 0, b \bmod p \neq 0$

因为，$ab \mid ap + bp + p^2$

所以，$ab \mid a + b + p$

假设 $a \leq b$，

设，$kab = a + b + p$

那么，$b=\frac{a+p}{ka-1}$

设，$d=ka-1$

那么，综上可得，$bd=a+p,a\mid d+1,a\le b$

分两种情况计算。

一种是$b\le d$，可以枚举$b$，上限为$\sqrt{p}$，再枚举$d$。通过$a=bd-p$，只考虑在取值范围内的$a$。

一种是$b>d$，可以枚举$d$，上限为$\sqrt{p}$，再枚举$b$。通过$a=bd-p$，只考虑在取值范围内的$a$。

- ③$a \bmod p=0,b \bmod p\neq 0$或$a \bmod p\neq 0,b \bmod p=0$

可以发现，②中满足条件的一组$(a,b)$，在这里对应着两组，分别是$(a,\frac{p(a+p)}{b})$和$(\frac{p(b+p)}{a},b)$。那么，这种情况下的方案数是②中方案数的两倍。

## 39.3 时空复杂度

时间复杂度：$O(T\sqrt{p})$

空间复杂度：$O(1)$

# 40. Hypertrees（HYPER）

## 40.1 题目大意

一个 3-超图类似于一个普通的图，只不过其中的边都连接三个点。
一个 3-超树是一个去掉任意一条边后都不连通的 3-超图。
给定 n，求有多少种含有 n 个带标号的点的本质不同的 3-超树。
共 T 组数据。
1≤T≤15，3≤n≤17

## 40.2 题目解法

因为数据范围较小，可以通过搜索求出结果。

可以枚举双连通分量的情况。对于同一双连通分量内的点，若看成从一两个点开始逐点扩展，那么每加入一条边就会加入一个新点。对于连接不同双连通分量的边，在双连通分量外部的作用与仅连接两个点的普通边相同。

搜索可能不能在时间限制范围内求出结果。
同样因为数据范围较小，可以打表。

## 40.3 时空复杂度

时间复杂度：$O(T)$

空间复杂度：$O(n)$

# 41. Max Circumference（MAXCIR）

## 41.1 题目大意

给出一个三角形 ABC，以及 n 个操作。第 i 个操作有两个参数 xi、yi，使用这个操作可以使得点 A 的 x 坐标增加 xi，y 坐标增加 yi。

可以使用最多 k 次操作，这些操作的影响叠加，同一个操作不能重复使用，ABC 三个点允许共线或重合。

求三角形 ABC 的最大周长，答案的绝对误差必须小于$10^{-12}$。

k≤n≤500

## 41.2 题目解法

|BC|不变，需要最大化|AB|+|AC|。

存在一个向量 x，AB、AC 在 x 方向上的投影长度之和最大时，|AB|+|AC|最大。在|AB|+|AC|最大时，对于过 A 点的以 B、C 为焦点的椭圆，A 点上的切线与 x 垂直。

按投影长度排序，排序结果发生变化的次数是$O(n^2)$的。枚举每一种情况，计算当时的最大值。

精度要求较高。在计算距离开平方前，用整数计算。开平方的计算，可以对结果的整数部分和小数部分分开计算来减小精度误差。

## 41.3 时空复杂度

时间复杂度：$O(n^2k)$

空间复杂度：$O(n^2)$

# 42. Trial of Doom

## 42.1 题目大意

约翰尼进入了一个大房间，这个房间被划分成了 n×m 个格子，每个格子可以是蓝色或红色。

现在约翰尼在（1,1），出口在（n,m），他需要到达终点并使得所有格子都是蓝色的。约翰尼可以移动到八个相邻的格子上，每当他离开一个格子，那么这个格子和它周围的四个格子会改变颜色。

现在给出房间的颜色情况，其中有 k 个格子是红色的，其他格子都是蓝色的。约翰尼想知道他是否能离开这个房间。

共 T 组数据。

$1 \leqslant T \leqslant 50$，$1 \leqslant n、m \leqslant 10^9$，$\min(n,m) \leq 40$，$k \leqslant 10000$

## 42.2 题目解法

假设 n＜m，如果 n=1，那么所有路径对颜色的影响都相同，直接判断就可以。

如果 n＞1，可以发现，无论哪些格子需要离开奇数次，都存在一种方案。那么只需要判断能否选出一些格子，修改它们以及它们周围四个格子的颜色，使所有格子变成蓝色。

若 m 较小，可以通过高斯消元来求出是否存在方案。

可以发现一个格子的情况对以后的列的影响是循环的，所以只需预处理出循环节，就可以快速列出方程。

## 42.3 时空复杂度

设循环节长度为 t。

时间复杂度：$O(Tnt+Tk)$

空间复杂度：$O(k+t)$

# 43. Sereja and Order（SEAORD）

## 43.1 题目大意

有 n 个程序，每个程序都需要在两台电脑上分别运行。第 i 个程序需要在第一台电脑上运行 A[i]秒，在第二台电脑上运行 B[i]秒。一台电脑不能同时运行两个程序，一个程序也不能同时在两台电脑上运行。

需要用最少的时间完成所有程序在两台电脑上运行的任务，求最短时间以及一种方案。

1≤n≤10000

## 43.2 题目解法

设所有程序在 A 电脑运行总时间是 $sum\_a$，在 B 电脑运行总时间是 $sum\_b$。

若存在一个程序 i，满足 $A[i]+B[i]\geq sum\_a+sum\_b$，那么最短时间就是 $A[i]+B[i]$。程序 i 在 A 电脑运行时，其他程序在 B 电脑运行。程序 i 在 B 电脑运行时，其他程序在 A 电脑运行。

否则，不妨设 $sum\_a\geq sum\_b$，那么最短时间就是 $sum\_a$。对于一个程序来说，它在 A、B 电脑上同时运行的概率较小，可以随机程序在 A、B 电脑上运行的先后顺序。在 A 电脑上不允许有空闲时间，所以每一个程序在 A 的运行时刻已经确定。程序在 B 的运行时刻可以通过贪心求出，每一个程序的运行时刻都是在不冲突的前提下尽可能早。需要验证程序在 B 的结束时刻是否不超过 $sum\_a$，来判断方案是否可行。

## 43.3 时空复杂度

设随机 t 次能够得到合法方案。

时间复杂度：$O(tn)$

空间复杂度：$O(n)$

# 44. Counting on a Tree（TREECNT2）

## 44.1 题目大意

给出一个包含 n 个点的树，边有边权，边权是不超过 m 的正整数。

计算有多少无序数对(S,T)，满足连接 S,T 两点的路径上，所有边权的最大公约数等于 1。

给出 q 次修改操作，一次修改操作修改一条边的边权。每一次操作后，都要再计算一次答案。

$1 \leqslant n \leqslant 10^5$，$m = 10^6$，$0 \leqslant q \leqslant 100$

## 44.2 题目解法

若$tot[i]$表示路径上的边权都是 i 的倍数的路径数，那么答案是$\sum_{i=1}^{m} tot[i] \cdot \mu[i]$。

需要计算所有的$tot[i]$。

若不修改，可以直接通过并查集求出。

因为修改次数较少，可以先将不会被修改的边加入并查集中。对于剩下的边，每一次计算时先将这些边加入，求出答案后再复原。所以不能进行路径压缩，可以启发式合并。

## 44.3 时空复杂度

设数据范围内数的约数个数不会超过 t。

时间复杂度：$O(m \log m + nt \log n + q^2 t \log n)$

空间复杂度：$O(nt + m \log m)$

# 45. Petya and Sequence（REALSET）

## 45.1 题目大意

给出一个长度为 n 的序列 A[0..n-1]，问是否存在一个长度为 n 的序列 B[0..n-1]满足：

1.至少存在一个 0≤i≤n-1，满足 B[i]≠0

2.对于任意 0≤j≤n-1，满足 $\sum_{i=0}^{n-1} A[i]\cdot B[(i+j)\bmod n]=0$ 。

## 45.2 题目解法

可以根据第二个条件，列方程求出序列 B 中元素的值。

方程组的系数矩阵 X 满足 $X[i][j]=A[(i+j)\bmod n]$。B 中所有元素都取 0 时满足第二个条件，因为第一个要求是不允许 B 中所有元素都取 0，所以判断是否有方案就是判断矩阵是否满秩。

设 $f(x)=\sum_{i=0}^{n-1} A[i]\cdot x^i$，若 $\gcd\left(f\left(x\right),x^n-1\right)$的次数为 d，那么这个矩阵的秩就是 $n-d$ 。

设 $\varphi_n(x)$ 满足 $\prod_{d|n}\varphi_d(x)=x^n-1$。可以发现，$\varphi_x(x)$ 无法继续分解。那么，只需判断是否存在 n 的约数 d 满足 $\varphi_d(x)\mid f(x)$。

这等价于 $x^d-1\mid f(x)\times\prod_{i\text{是d的质因子}}(x^{d/i}-1)$。可以模拟来判定。

## 45.3 时空复杂度

设 n 的约数个数为 t。

时间复杂度：$O(n\log n+nt)$

空间复杂度：$O(n)$

# 46. Game of Numbers（GNUM）

## 46.1 题目大意

给出两个长度为 n 的数列 A[1..n]和 B[1..n]。有两个集合 S1、S2，初始都为空。

一次操作要选出两个二元组(i,j)和(p,q)，满足

- (i,j)不在 S1 中，(p,q)不在 S2 中
- A[i]<B[j]，B[p]<A[q]
- $\gcd(A[i],B[j])\neq 1$，$\gcd(A[q],B[p])\neq 1$
- $\gcd(\gcd(A[i],B[j]),\gcd(A[q],B[p]))\neq 1$

然后将(i,j)加入 S1 中，将(p,q)加入 S2 中。

求最多进行多少次操作。

共 T 组数据。

1≤T≤10，1≤n≤400，1≤A[i]、B[i]≤$10^9$

## 46.2 题目解法

对于(A[x],B[y])，若 $\gcd(A[x],B[y])\neq 1$，根据它们的大小关系，可以判断应属于 S1 还是 S2。

分别求出可以加入 S1、S2 的数对。若属于 S1 的(A[x1],B[y1])和属于 S2 的(A[x2],B[y2])满足 $\gcd(A[x1],A[x2],B[y1],B[y2])\neq 1$，则它们可以在同一次操作中被选择。这样可以通过二分图最大匹配或最大流求出最大操作次数。

但这样边数太多。可以对于所有在 $\gcd(A[x],B[y])$ 中出现过的质因子建立一个结点，数对之间不连边，而是数对(A[x],B[y])与 $\gcd(A[x],B[y])$ 的所有质因子连边。这样与原来的图是等价的。

## 46.3 时空复杂度

时间复杂度：$O(Tn^6\log^2 n)\langle$在有$n^2$个点$n^2\log n$条边的图上求最大流$\rangle$

空间复杂度：$O(n^2\log n)$

# 47. Billboards（BB）

## 47.1 题目大意

公路上有 n 个广告牌，大厨要选择一部分广告牌放广告。为了加深对路人的影响，任意连续 m 个广告牌中至少要放 k 个广告。

大厨希望在满足条件的同时使用最少的广告牌。

大厨想知道有多少种使用广告牌的方案。

共 T 组数据。

1≤T≤300，1≤k≤m≤500，m≤n≤$10^9$

## 47.2 题目解法

先考虑 m|n 的情况。n 个广告牌 m 个一组进行分组，每组放 k 个。可以把放的位置写成一个 $\frac{n}{m}$ 行 k 列的矩阵，第 i 行第 j 列的数 X[i][j]表示第 i 组中的第 j 个广告放在这一组中的第 X[i][j]个广告牌上。那么为了满足要求，矩阵中每一行的数单调递增，每一列的数单调不下降。

这种矩阵是杨氏矩阵，可以用钩子公式计算方案数：

$$ans=\prod_{(i,j)}\frac{r+j-i}{hook(i,j)}$$

其中 r 表示可选数集大小。hook(i,j)表示同一行右方的格子数，加上同一列下方的格子数，再加上 1，表示它自己。

直接计算的计算量太大。可以发现分子、分母上重复的数很多，约分重复的数，剩下的数就比较少了。

再考虑 m 不能整除 n 的情况。

如果 $n \bmod m \le m-k$，那么最后多出来的一组中一个广告也不放，为了满足要求，矩阵中的数都要比 $n \bmod m$ 大。这相当于减小了可选数集的大小。

如果 $n \bmod m > m-k$，那么矩阵中每一行的最后 $m-n \bmod m$ 个数必须是 $n \bmod m+1$ 到 $m$。这相当于减小了矩阵的列数。

那么，m 不能整除 n 的情况就都可以变成 m 整除 n 的情况来做。

## 47.3 时空复杂度

时间复杂度：$O(Tmk)$

空间复杂度：$O(1)$

# 48. Graph Challenge（DAGCH）

## 48.1 题目大意

给出一个 n 个点 m 条边的有向图。结点按 dfs 序编号。

x 对于 y 来说是好的，当且仅当存在一条路径 x、v0、v1、v2、……、y，满足 x＜y＜v0、v1、v2……。

对于 y 来说是好的点中，编号最小的点 x，称 x 对于 y 来说是最好的。

有 q 次询问，每次询问一个点，求它对于多少点来说是最好的。

共 T 组数据。

$1 \leqslant T \leqslant 10$，$2 \leqslant n \leqslant 10^5$，$n-1 \leqslant m \leqslant 2 \times 10^5$，$1 \leqslant q \leqslant 10^5$

## 48.2 题目解法

根据题目中的描述，可以发现，若按 dfs 序进行求支配树的算法，那么对于一个点来说最好的点，就是它的半必经点。

一次询问，就是求一个点是多少点的半必经点。

## 48.3 时空复杂度

时间复杂度：$O(T(n+m+q))$

空间复杂度：$O(T(n+m))$

# 49. Counting on a directed graph（GRAPHCNT）

## 49.1 题目大意

给定一个 n 个点 m 条边的有向图。统计无序对(X,Y)的个数，其中(X,Y)满足存在一条从点 1 到点 X 的路径，和一条从点 1 到点 Y 的路径，且两条路径除了点 1 以外没有公共点。

$1 \leqslant n \leqslant 10^5$，$0 \leqslant m \leqslant 5 \times 10^5$

## 49.2 题目解法

可以发现，点对(X,Y)是合法的，当且仅当从点 1 出发，X 和 Y 没有除了点 1 以外的公共必经点。

以点 1 为根，求出图的支配树，那么点对(X,Y)合法，当且仅当它们的 LCA 是点 1。

求出每个子树大小，枚举点 1 的子结点就可以求出答案。

## 49.3 时空复杂度

时间复杂度：$O(n+m)$

空间复杂度：$O(n+m)$

# 50. Queries With Points（QPOINT）

## 50.1 题目大意

给出 n 个简单多边形，多边形之间不相交。
给出 q 次询问，每次询问一个点在哪个多边形内部，或不在任何多边形内部。
强制在线。

$1 \leqslant n \leqslant 10^5$，$1 \leqslant q \leqslant 10^5$

所有多边形顶点个数和 $k \leqslant 3 \times 10^5$

## 50.2 题目解法

若不强制在线，可以将询问按横坐标排序。顶点的横坐标作为关键点，相邻关键点之间线段的上下关系不变。随着横坐标从小到大，可以用平衡树维护线段的上下关系。

因为强制在线，所以可以用可持久化平衡树。

## 50.3 时空复杂度

时间复杂度：$O(k \log k + q \log k)$

空间复杂度：$O(k \log k)$

# 51. Fibonacci Numbers on Tree（FIBTREE）

## 51.1 题目大意

给出一棵 n 个结点的树，结点有点权，初始都是 0。

支持这几种操作：

• A x y：在 x 到 y 的路径上，第 1 个点加上 fib[1]，第 2 个点加上 fib[2]，第 3 个点加上 fib[3]……其中 fib[i]是斐波那契数列的第 i 项，令 fib[1]=1，fib[2]=1。

• QS x y：求以 x 为根时，y 的子树中所有结点的点权和。

• QC x y：求 x 到 y 的路径上所有结点的点权和。

• R x：所有结点的点权还原到第 x 个操作后的状态。

共 m 次操作。

输出时对$10^9+9$取模。

强制在线。

$1\leqslant n\leqslant 10^5$，$1\leqslant m\leqslant 10^5$

## 51.2 题目解法

对树进行树链剖分，路径和子树的问题都变成了序列上的问题。

斐波那契数列的通项公式是，$fib[i]=\frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^i-\left(\frac{1-\sqrt{5}}{2}\right)^i\right)$。

存在一个整数 x，满足$x^2\equiv 5(\mathrm{mod}10^9+9)$，可以用它代替$\sqrt{5}$。

那么，A 操作相当于对一个区间加上两个等比数列的差，QS、QC 操作相当于对一个区间求和。这通过带标记的线段树可以维护。

R 操作需要记录历史信息，可以把线段树改成可持久化线段树。

## 51.3 时空复杂度

时间复杂度：$O(n+m\log^2 n)$

空间复杂度：$O(n+m\log^2 n)$

# 52. Random Number Generator（RNG）

## 52.1 题目大意

已知数列A[i]的前k项A[1]、A[2]、A[3]、……、A[k]。
数列递推公式：

$$A[i]=\left(A[i-1]\cdot C[1]+A[i-2]\cdot C[2]+\ldots A[i-k]\cdot C[k]\right)\bmod 104857601$$

给出C[1]、C[2]、C[3]、……、C[k]。
求A[n]。

$1\leqslant n\leqslant 10^{18}$，$1\leqslant k\leqslant 30000$

## 52.2 题目解法

数列递推公式是线性递推式，可以通过矩阵乘法优化计算。

可以发现，这个矩阵的特征多项式是 $f(x)=\sum_{i=1}^{k}C[i]\cdot x^{i-1}$，可以通过特征多项式进一步优化。

可以证明，若 $g(x)=x^n \bmod f(x)$，将 $g(x)$ 表示成 $g(x)=\sum_{i=1}^{k}B[i]\cdot x^{i-1}$ 的形式，那么，$A[n]=\sum_{i=1}^{k}B[i]\cdot C[i]$。

求 $g(x)$ 时，可以通过快速幂和FFT进行优化。

## 52.3 时空复杂度

时间复杂度：$O(k\log k\log n)$

空间复杂度：$O(k)$

# 53. Chefbook（CHEFBOOK）

## 53.1 题目大意

有一个 n×n 的矩阵 L。有 m 个要求，一个要求是使 L 中某个位置上的数在一个范围内。

可以对矩阵进行修改。修改方式是使某一行的数都加上一个正数，或使某一列的数都减去一个正数。

在满足所有要求的前提下，求矩阵中被限制取值范围了的数的和的最大值。

1≤n≤100

## 53.2 题目解法

一个数的值是原值加上改变量。原值已经确定，所以最大化值的总和，就是最大化每个数改变量的总和。

用变量 $x_i$ 表示某一行或某一列的修改量，用 $c_i$ 表示一共有多少数进行了 $x_i$ 的修改。问题就变成了最大化 $\sum x_i c_i$ 。

对一个数取值范围的限制，可以写成 $x_i - x_j \le k$ 的形式。

可以通过最小费用最大流求解。每一个变量对应着一个点，对于每一个形如 $x_i - x_j \le k$ 的限制，在 i、j 之间连接一条流量是正无穷、费用是 k 的边。若 $x_i$ 是对某一行的修改，那么它是正数，从源点 S 向 $x_i$ 连接一条流量是 $c_i$、费用是 0 的边。若 $x_i$ 是对某一列的修改，那么它是负数（减去一个正数看成是加上一个负数），从 $x_i$ 向汇点 T 连接一条流量是 $c_i$、费用是 0 的边。

具体的方案可以通过差分约束系统构造出来。

## 53.3 时空复杂度

时间复杂度：$O(nm^2)$

空间复杂度：$O(n+m)$

# 54. Short（SHORT）

## 54.1 题目大意

给出两个数 n，k。找出所有的数对(a,b)，满足：

- n<a<k
- n<b<k
- ab-n 可以被(a-n)(b-n)整除

$0 \leqslant n \leqslant 10^5$，$k \leqslant 10^{18}$

## 54.2 题目解法

用 a 替换 a-n，用 b 替换 b-n，限制条件变成：

- 0<a<k-n
- 0<b<k-n
- $ab \mid (a+n)(b+n)-n$

可以得到，

$$ab \mid ab+an+bn+n^2-n$$

$$ab \mid an+bn+n^2-n$$

设，$kab = an+bn+n^2-n$

那么，$b = \dfrac{an+n^2-n}{ka-n} = \dfrac{n(a+n-1)}{ka-n}$

不妨设 a<b，那么可以从 1 开始枚举 a，枚举次数是 0(n)的。

当 a 较小时，$n(a+n-1)$ 较小，可以枚举它的约数，验证能否作为 $ka-n$。

当 a 较大时，可以发现直接枚举 k 就可以，枚举次数不会太多。

## 54.3 时空复杂度

时间复杂度：$O(n\log n+nt)$,$t$表示对于一个$a$的枚举量

空间复杂度：$O(n\log n)$

# 55. Selling Tickets（TICKETS）

## 55.1 题目大意

有 n 道菜，m 位顾客。每一位顾客都有两道特别喜爱的菜，而只要吃到了至少一道他喜爱的菜，这位顾客就会感到很高兴。每道菜最多只能供应给一位顾客。

厨师想要卖出尽可能多的门票，但同时要能够保证，无论哪些顾客购买门票，所有到来的顾客都能感到高兴。

求厨师最多能够卖多少门票。

共 T 组数据。

$1 \leqslant T \leqslant 15$，$2 \leqslant n \leqslant 200$，$0 \leqslant m \leqslant 500$

## 55.2 题目解法

可以把菜看做点，一位顾客喜爱两道菜，可以看做边。这样问题就变成了，求最大的 t，满足任选 t 条边，都能够为每条边分配一个不同的点，并且分配到的点是边的端点之一。

直接求 t 不太方便，难以验证所有情况。

可以反过来求，求最小的不合法的 t，那么 t-1 就是最大的合法的。

那么，就是求边数恰好比点数多一的联通子图，点数最少是多少。

可以分两种情况：

- 两点之间的三条路径。

这种情况可以枚举这两个点，然后通过 BFS 求解。

- 两个有一条公共边的简单环。

这种情况可以枚举两个环相交位置的点，以它为根作 BFS 树。一个简单环对应树上一条路径。把每一个环的权值设为它的边数再加上它两端的 LCA 到根的距离，用此时权值最小的两个环的权值之和尝试更新答案。

## 55.3 时空复杂度

时间复杂度：$O(n^2m)$

空间复杂度：$O(n+m)$

# 56. Luckdays（LUCKYDAY）

## 56.1 题目大意

定义一个数列 S，其中，

$S[1]=A$，$S[2]=B$，$S[i]=(X\cdot S[i-1]+Y\cdot S[i-2]+Z)\bmod P, i\geq 3$

给出 A、B、X、Y、Z、P。其中 P 是质数。

有 Q 次询问。第 i 次询问是，有多少 k 满足 $L[i]\leq k\leq R[i]$，且 $S[k]=C$。这里的 C 对于每一次询问都是相同的。

2≤P≤10007，1≤Q≤20000，1≤L[i]、R[i]≤$10^{18}$

## 56.2 题目解法

S 是循环的。X、Y 中有 0 时，循环节长度是 $O(P)$ 的，可以暴力求出循环节长度和每个值的出现位置。

X、Y 都不是 0 时，循环节长度是 $O(P^2)$ 的。可以通过大步小步求出循环节长度和一个循环节内部 $S[k]=C$ 的所有出现位置。

对于每次询问，完整的循环节可以直接计算，不完整的部分可以二分出现了多少次。

## 56.3 时空复杂度

时间复杂度：$O(P\sqrt{P}+Q\log P)$

空间复杂度：$O(P\sqrt{P})$

# 57. Cool Numbers（COOLNUM）

## 57.1 题目大意

对于一个数 x，设它各个数位上的数之和为 T，若存在一种方案，从中选出一到三个数位，这几个数位上的数之和为 S，且$(T-S)^S$是 x 的倍数，那么称 x 是一个 cool number。

给出一个数 n，求小于等于 n 的最大的 cool number，和大于 n 的最小的 cool number。

共 T 组数据。

$1\leqslant T\leqslant 10^5$，$1\leqslant n\leqslant 10^{1000}$，所有 n 的长度之和 $m\leqslant 4\times 10^6$

## 57.2 题目解法

对于是 cool number 的 x，分两种情况考虑。

• x 的各个数位中，非零的数位不超过 3 个。

选出这几个非零数位，它们的和作为 S，那么 S=T，$(T-S)^S=0$，0 一定是 x 的倍数。所以这类数都是 cool number。

可以通过贪心来求出这种情况下的答案。

• x 的各个数位中，非零的数位超过 3 个。

这时$T-S>0$。可以发现，x 的位数不会超过 77。可以先枚举$T-S$，再枚举$(T-S)^S$，然后验证枚举到的数是否是 cool number。

将求出的 cool number 排序。对于每次询问，通过二分得到这种情况下的答案。

## 57.3 时空复杂度

设第二类 cool number 有 t 个。

时间复杂度：$O(\log t+m)$

空间复杂度：$O(t)$

# 58. Chef and Tree Game（GERALD08）

## 58.1 题目大意

大厨有一棵有根树 G，其上有 n 个节点，树上的边要么是黑色，要么是白色。

一次大厨的朋友 Ksen 提议在树上玩一个游戏，规则如下：

两名玩家轮流行动。大厨每一次删除一条还在树上的黑边，Ksen 每次删去一条还在树上的白边。当一条边被删去之后，所有与根结点不连通的边也会一起被删去。当一名玩家无法行动时，他就输了。

他们都按照最优策略行动，求大厨先手时谁会赢、Ksen 先手时谁会赢。

$1 \leqslant n \leqslant 10^5$

## 58.2 题目解法

可以为结点定义一个局面函数 $f$ 。

一个结点的函数值是所有子结点的贡献之和。

若结点 i 连向其父结点的边是黑色的，令 a 为满足 $f_i + a > 1$ 的最小正整数，那么它对父结点的贡献是 $\frac{f_i + a}{2^{a-1}}$ 。

若结点 i 连向其父结点的边是白色的，令 a 为满足 $f_i - a < -1$ 的最小正整数，那么它对父结点的贡献是 $\frac{f_i - a}{2^{a-1}}$ 。

特殊地，叶结点的函数值为 0。

记根结点的函数值为 x。若 x 为 0，那么后手必胜。若 x 为非零整数，那么大厨必胜。若 x 不是整数，那么 Ksen 必胜。

求函数值可以用高精度实现。

## 58.3 时空复杂度

时间复杂度：$O(n \log^2 n)$

空间复杂度：$O(n \log n)$

# 59. Attack of the Clones（CLONES）

## 59.1 题目大意

我们称一个形为 f:A→B 的函数为布尔函数，其中 A 是所有长度为 n 且仅由 0 和 1 组成的数列的集合，B={0,1}，我们称 n 为布尔函数的项数。

有四种特殊的布尔函数集合：

- Z 是 0-保留值函数集合：满足 f(0,0,...,0) = 0
- P 是 1-保留值函数集合：满足 f(1,1,...,1) = 1
- D 是自对偶函数集合：满足!f(x1,x2,...,xn) = f(!x1,!x2,...,!xn)
- A 是仿射函数集合：

  如果 $f(x_1,x_2,...,x_{i-1},a,x_{i+1},...,x_n)=f(x_1,x_2,...,x_{i-1},b,x_{i+1},...,x_n)$

  那么 $f(y_1,y_2,...,y_{i-1},a,y_{i+1},...,y_n)=f(y_1,y_2,...,y_{i-1},b,y_{i+1},...,y_n)$

  这对任意 i、a、b、x、y 都成立

给出由 Z、P、D、A、!、v、^、\、(、)组成的表达式。!表示补集，v 表示并集，^表示交集，\表示差集。(、)优先级最高，其次是!，v、^、\的优先级相同。这个表达式表示了一个布尔函数集合，求这个集合的大小。

共给出 m 个表达式，表达式的长度不超过 S。

1≤n、m≤100，1≤S≤100

## 59.2 题目解法

根据函数是否属于 Z、P、D、A，可以把函数分为 16 类，一类函数同时出现或同时不出现在表达式表示的集合中。

根据表达式，求出每一类函数是否属于表达式表示的集合中。共 16 类函数，所以可以用$[0,2^{16})$中的数表示一个集合。这个过程可以借助栈或递归实现。

Z、P 是确定了函数在一处的取值。

D 是将函数每两处的取值分为一组，一组内两处取值相反。那么，一组内确定一处的值就可以，相当于定义域减小到原来的一半。

A 是对于每一个位置 i，$f(x_1,x_2,...,x_i,...,x_n)$与$f(x_1,x_2,...,!x_i,...,x_n)$取值相同还是不相同，对所有自变量数列都一样。这相当于，只要确定了自变量数列中每一个位置变化后函数值是否改变，只需再确定函数在一处的取值，整个函数就确定了。

根据这些，能够求出每一类函数的数量与 n 的关系。

## 59.3 时空复杂度

时间复杂度：$O(mS^2)$

空间复杂度：$O(S)$

# 60. Colored Domino Tilings and Cutsontest（DOMNOCUT）

## 60.1　题目大意

一个棋盘覆盖的染色是指：在棋盘上填上小写字母，使得每个格子有且仅有一个相邻格子的字母和他的一样。一个格子与另一个格子相邻当且仅当他们有公共边。每个字母对应一种颜色。

棋盘的割是指一条竖直或水平的直线将棋盘分成两半，而且这两半都是合法的棋盘覆盖染色。也就是说，这条直线不能穿过一对有相同颜色的相邻格子。

给出棋盘边长 n 和 m，构造一个棋盘覆盖染色，在最小化棋盘的割的数量的前提下，最小化使用的颜色的数量。

共 T 组数据。

1≤T≤1000，1≤n、m≤500

## 60.2　题目解法

这类似于多米诺骨牌覆盖，有相同颜色的相邻格子被同一块多米诺骨牌覆盖。割就是不穿过多米诺骨牌的水平或竖直的直线。染色时要求相邻多米诺骨牌颜色不同。

格子数量 $n\times m$ 是奇数时一定无解，否则一定有解。

当 n 或 m 较小时，染色方式比较容易确定，可以分类打表。

当 n、m 较大时，明显没有割的数量少且仅用两种颜色的方案。可以构造出 $5\times6$ 和 $6\times8$ 两种棋盘，它们的割的数量是 0，使用了 3 种颜色，并且它们可以在不改变割的数量的情况下增加两行或者两列。通过这两种特殊的棋盘，可以构造出 n、m 较大时的方案，割的数量仍然是 0，仍然使用了 3 种颜色。

## 60.3　时空复杂度

时间复杂度：$O(Tnm)$

空间复杂度：$O(nm)$

# 61. Easy Exam（EASYEX）

## 61.1 题目大意

有一个K面的骰子，每个面上的数字分别是1到K。给出两个参数L和F。将这个骰子掷N次，令$a_i$为掷出数字i的次数。求$\prod_{i=1}^{L} a_i^{\,F}$的期望值。

1≤N、K≤$10^9$，1≤$L\cdot F$≤20000，1≤F≤1000

## 61.2 题目解法

用$x_{i,j}$表示第j次掷骰子掷到的是不是i。如果是，$x_{i,j}=1$。否则$x_{i,j}=0$。

那么，$\prod_{i=1}^{L} a_i^{\,F}=\prod_{i=1}^{L}\left(\sum_{j=1}^{N} x_{i,j}\right)^F$

将式子展开，每一项单独计算。

对于一项，若存在两个不同的数s、t和一个数c，使$x_{s,c}$和$x_{t,c}$的指数都大于0，那么这一项的值恒为0，因为$x_{s,c}$和$x_{t,c}$不可能同时为1。若不存在，如果指数大于0的$x_{i,j}$有tot个，那么这一项的期望值是系数乘上$\frac{1}{K^{tot}}$。

考虑对每一个tot值，计算这些项的系数和。这可以通过dp求出。

用$f[n][m]$表示将$\prod_{i=1}^{n}\left(\sum_{j=1}^{N} x_{i,j}\right)^F$展开后tot值为m的项的系数和，用$h[n][m]$表示$\left(\sum_{i=1}^{N} x_i\right)^n$展开后tot值为m的项的系数和，那么，

$$f[n][m]=\sum_{t=0}^{m} f[n-1][t]\cdot h[F][m-t]\cdot C_{N-t}^{m-t}$$

令$g[n][m]=f[n][m]\cdot(N-m)!$，那么

$$g[n][m]=\sum_{t=0}^{m} g[n-1][t]\cdot\frac{h[F][m-t]}{(m-t)!}$$

这可以用快速幂和FFT优化。

其中$h[n][m]$的计算方式是，

$$h[n][m]=h[n-1][m]\cdot m+h[n-1][m-1]\cdot(N-m+1)$$

计算出 $g[n][m]$，进而计算出 $f[n][m]$，就可以计算答案了。

## 61.3 时空复杂度

时间复杂度：$O(LF\log LF\log L+F^2)$

空间复杂度：$O(LF)$

# 62. Little Party（LPARTY）

## 62.1 题目大意

Leha 有 n 个好朋友，他和这 n 个朋友经常聚会，但有时聚会不愉快。

共有 m 次不愉快的聚会，Leha 记录了那些时候每个朋友的心情。用字母表示朋友，大写是心情好，小写是心情不好。例如 n=3 时，AbC 表示第二个朋友心情不好，其他两个朋友心情好。

Leha 想总结一些一定会带来问题的子集。例如 Ab 表示只要出现 Ab，聚会就会不愉快，那么这 m 次记录中必须同时有 AbC 和 Abc。

Leha 想用一些子集表示所有这 m 次记录。例如记录有 AbC、Abc、aBC，可以用子集 Ab、aBC 表示，这时子集长度和为 5。当然也可以用子集 AbC、Abc、aBC 表示，但这时子集长度和为 9。

求子集长度和的最小值。

共 T 组数据。

1≤T≤120，1≤n≤5，0≤m≤1000

## 62.2 题目解法

共有 $3^n$ 种子集，n=5 时，$3^n = 243$。搜索选择哪些子集。

需要加一些剪枝。

最重要的剪枝就是如果集合 A 包含于集合 B，那么如果 A 合法，就不需要考虑选 B。因为 A 的作用比 B 大，并且长度比 B 小。

还可以加一些其他剪枝。

如果当前总长度已经超过之前搜索到的最优总长度，不必再搜索。

如果剩余集合全部选择也不能覆盖所有 m 次记录，不必再搜索。

先考虑长度小的集合，再考虑长度大的集合。

搜索过程中可以用位运算优化状态的记录。

## 62.3 时空复杂度

时间复杂度：$O(T2^{3^n})$

空间复杂度：$O(3^n + m)$

# 63. Two Roads（TWOROADS）

## 63.1 题目大意

在平面直角坐标系中，有 n 个互不重合的点，给出点的坐标。

现在需要作两条直线。对于一个点，设它到两条直线中较近的直线的距离为 d，那么这个点的权值是 $d^2$。

选择合适的两条直线，最小化所有点的权值之和。求权值和最小值。

3≤n≤100

## 63.2 题目解法

两条直线夹角的角平分线将平面划分为四部分，每条直线控制其中不相邻的两部分。

可以发现角平分线是互相垂直的，并且其中一条角平分线过两个点，另一条角平分线过一个点。

枚举过两个点的角平分线，再枚举另一条角平分线，就得到了划分平面的方案，那么每一个点的权值是由哪一条直线计算出来的就确定了。

对每一条直线，通过由它控制的点，计算那些点的最小权值和。

## 63.3 时空复杂度

时间复杂度：$O(n^3 \log n)$

空间复杂度：$O(n)$

# 64. Expected Maximum Matching（MATCH）

## 64.1 题目大意

有一个二分图，左边有 n 个点，右边有 m 个点。$f[i][j]$表示左边第 i 个点与右边第 j 个点之间有边的概率。

给出 n、m、$f[i][j]$，求这个二分图的最大匹配的期望值。

1≤n≤5，1≤m≤100，0≤$f[i][j]$≤1

## 64.2 题目解法

由于 n 比较小，可以考虑状态压缩动态规划。

令$f[i][j]$表示只考虑右边前 i 个点时，左边点集的每一个子集能否全部与右边匹配。左边有 n 个点，就有$2^n$个点集。每一个点集用 0 或 1 表示能否全部与右边的点匹配，所有点集的信息存储在 j 的每一个二进制位上。

这样 j 会有$2^{2^n}$个，但实际上会出现的 j 不会有太多。如果点集 A 是点集 B 的子集，那么点集 B 中的点能全部匹配时，点集 A 中的点必定可以全部匹配。当$n=5$时，会出现的 j 不到 10000 个。

$f[i][j]$向$f[i+1][k]$转移时，枚举右边的第 i+1 个点与左边的哪些点之间有边。对于点集 A，若存在左边一个点 S，满足$S\in A$，并且 S 与右边的第 i+1 个点之间有边，那么如果$A\backslash\{S\}$在 j 中是 1，A 在 k 中就是 1。当然，如果 A 在 j 中就是 1，那么无论如何 A 在 k 都是 1。

当 j 确定，并且右边新加入的那个点与左边的点的连接情况确定时，k 就确定了。可以预处理出每一个 j、每一种连接情况下，会转移到的 k。

根据$f[m][j]$的值，可以计算出最大匹配的期望值。

## 64.3 时空复杂度

设 s 为实际上会出现的状态的数量。

时间复杂度：$O\left(2^{2n}ns+2^nms\right)$

空间复杂度：$O(ms)$

# 65. Ciel and Earthquake（CIELQUAK）

## 65.1 题目大意

有一个 n×m 的网格，有公共边的网格之间有一条双向道路。每一条道路都有 p 的概率损坏。求(1,1)和(n,m)连通的概率。

共 T 组数据。

$1 \leqslant T \leqslant 50$，$1 \leqslant n \leqslant 8$，$1 \leqslant m \leqslant 10^{18}$，$0.1 \leqslant p \leqslant 1$

## 65.2 题目解法

m 较小时，可以通过轮廓线动态规划求解。记录轮廓线上的点之间的连通性，以及它们与(1,1)的连通性。在 $n=8$ 时，会出现的状态数 S 只有几千个。

m 较大时，可以通过 m 较小时的计算结果近似地计算 m 较大时的结果。假设 $Ans(n,m)$ 表示 n×m 的网格的答案，那么，

$$Ans(n,m) \approx Ans(n,x) \cdot \left( \frac{Ans(n,x+1)}{Ans(n,x)} \right)^{m-x}$$

这里，x 的值越大误差越小，当 x 取 35 左右时，就能够达到精度要求。

## 65.3 时空复杂度

时间复杂度：$O(TnxS)$

空间复杂度：$O(nxS)$

# 66. Counting Hexagons（CNTHEX）

## 66.1 题目大意

有 n 种木棍，长度分别是 1 到 n，每种木棍有 k 根。

选出六根木棍拼成一个面积为正的六边形，要求最长的木棍长度至少为 L，其他木棍长度不能超过 X。

求方案数。

$2\leqslant L\leqslant n\leqslant 10^9$，$0\leqslant n-L\leqslant 100$，$1\leqslant X<L$，$1\leqslant k\leqslant 5$

## 66.2 题目解法

拼出的六边形面积为正，当且仅当最长的木棍的长度小于其他木棍的长度和。

由于 n-L 较小，可以枚举最长的木棍的长度，计算此时的方案数。

可以通过数位 dp 求解。

$f[a][b][c][d][e]$表示在后 a 位中，其他木棍的长度和与最长木棍长度的大小关系是 b，与 X 的大小关系是 c，需要向第 a+1 位的进位为 d，它们之间的相等关系是 e，这时的方案数。

$f[a]$向$f[a+1]$转移时需要枚举第 a+1 位的数值。

## 66.3 时空复杂度

时间复杂度：$O((n-L)\log n)$

空间复杂度：$O(\log n)$

# 67. Something About Divisors（DIVISORS）

## 67.1 题目大意

给出正整数 B 和 X，求有多少正整数 N 满足，至少存在一个数 D（N<D<=B）能整除 $N \cdot X$ 。

共 T 组数据。

1≤T≤40，1≤B≤$10^{12}$，1≤X≤60

## 67.2 题目解法

设 $v = \frac{NX}{D}$，那么 $v \mid NX$，$\frac{v}{\gcd(v, X)} \mid N$ 。

所以，$N = k \cdot \frac{v}{\gcd(v, X)}$ 。

可以枚举 v，计算此时 N 的数量。由于 $N \le \frac{Bv}{X}$，所以 $k \le \frac{B \cdot \gcd(v, X)}{X}$ 。那么此时 N 的数量就是 $\frac{B \cdot \gcd(v, X)}{X}$ 。

一个 N 可能被多个 v 计算。可以限制每一个 N 都只在最大的合法的 v 处计算。

这可以通过容斥来求出。直接容斥计算量太大，需要加一些优化。容斥时需要计算当前枚举到的状态的最小公倍数，可以将最小公倍数相同的合并在一起计算。如果已经确定当前枚举到的这一类状态对答案的贡献是 0，就不必继续计算。

## 67.3 时空复杂度

设进行一次容斥的计算量是 tot。

时间复杂度：$O(TXtot)$

空间复杂度：$O(Xtot)$

# 68. Minesweeper Reversed（MINESREV）

## 68.1 题目大意

给出一个 n×m 的扫雷棋盘，给出其中雷的位置。

最初所有格子都是打开的，你需要关闭所有的格子。

你可以通过点击一次来关闭一个格子。在关闭(x,y)后，在正常扫雷规则中可能与(x,y)同时打开的格子也会被关闭。

求最少点击多少次可以关闭所有的格子，包括有雷的格子。

共 T 组数据。

1≤T≤50，1≤n、m≤50

## 68.2 题目解法

有雷的格子一定需要一次点击，且不能和其他格子在同一次点击中被关闭。所以只需要考虑没有雷的格子。

将这些格子分为两类，一类是周围有雷的格子，另一类是周围没有雷的格子。其中第二类格子形成一个个连通块。

不与连通块相邻的第一类格子需要一次点击来关闭，且不能和其他格子在同一次点击中被关闭。

存在一种方案关闭剩下的格子，就是每个连通块都点击一次。需要点击的次数是连通块的数量。但这样不一定是最优的，因为可能存在一些第一类格子，在点击它的时候可以同时关闭多个连通块。

可以发现，与连通块相邻的第一类格子至多与两个连通块相邻。

可以把连通块看做点，与两个连通块相邻的第一类格子看做边，连接它相邻的连通块对应的点。这样可以得到一个图。那么，连通块的数量减去这个图的最大匹配就是点击的最小次数。

一般图的最大匹配可以通过带花树求出。

## 68.3 时空复杂度

时间复杂度：$O(Tn^2m^2)$

空间复杂度：$O(nm)$

# 69. To Queue or not to Queue（TMP01）

## 69.1 题目大意

有一个字符串 S，初始为空。会有两种操作，

- 在 S 的末尾插入一个字符
- 删除 S 的第一个字符

每次操作后，都求一遍当前的 S 有多少个不同的子串。

共 n 次操作。

$n \leqslant 10^6$

## 69.2 题目解法

如果没有删除操作，可以直接用 Ukkonen 算法维护后缀树。

有删除操作，相当于是每次删除一个最长的后缀。

记录每一个后缀的插入位置，删除时删除这个插入位置所在结点，合并度数为 1 的祖先结点。

## 69.3 时空复杂度

时间复杂度：$O(n)$

空间复杂度：$O(n)$

# 70. Two Magicians（MAGIC）

## 70.1 题目大意

给出一张 n 个点，m 条边的图。最初第一个人在 1 号点，第二个人在 2 号点。两个人轮流操作，操作分三个步骤：

•可以沿着现有的无向边移动任意步，如果这一步结束时两个人在同一个格子，则当前人胜。

• 加入一条图中没有的边，若无法加入，则另一个人胜。

•最开始每个人都有 p 次传送机会。若当前人还有传送机会，可以消耗一次机会传送到任意一个结点。

求必胜的是先手还是后手。

共 T 组数据。

1≤T≤100，2≤n≤7777，0≤m≤10000，0≤p≤10000

## 70.2 题目解法

题目描述的操作方式等价于如下方式，

• 若两人在同一连通块，当前人胜。

• 加入一条图中没有的边。

• 若当前人还有传送机会，可以消耗一次机会传送到任意一个连通块。

对游戏结果有影响的状态有：两人所在连通块的点数的奇偶性、点数为奇数、偶数的连通块的数量、加入后不会影响连通性的边数、两人剩余的传送机会数。

以这些为状态，可以进行动态规划。但动态规划复杂度较高。可以发现，当连通块数较多时，dp 值会出现循环或不变的情况。那么，可以在较小范围内进行动态规划，对于较大的情况，可以根据较小范围内的 dp 值得出答案。

## 70.3 时空复杂度

时间复杂度：$O(n+m)$

空间复杂度：$O(n)$

# 71. Simple Queries（DISTNUM）

## 71.1 题目大意

给出一个长度为 n 的数列，支持这几种操作，

• 1 l r：令 S 为由下标范围从 l 到 r 的不同的元素构成的有序集合，求，

$$\left(\sum_{1\le i<j<k\le|S|} S_i S_j S_k\right)\left(\mathrm{mod}\,10^9+7\right)$$

• 2 x y：将第 x 个数修改成 y

• 3 x：删除第 x 个数

• 4 x y：在第 x 个数后面插入 y

• 5 x y：求下标范围从 l 到 r 的区间中有多少不同的元素

共 m 次操作。

$1\leqslant n、m\leqslant 10^5$

## 71.2 题目解法

可以对每一个数记录下一个相同的数的出现位置 next[i]。

对于询问 5，就是求区间[l,r]中有多少位置的 next[i]比 r 大。

对于询问 1，若 S 中的数的和是 A，平方和是 B，三次方和是 C，那么答案就是 $\frac{A^3-3AB+2C}{6}$。

这些都可以通过树套树来实现。

为了方便，可以预先为所有出现过的数分配一个位置，当这个数不在数列中时，可以令它的值为 0。这样就不必考虑位置的变化了。

## 71.3 时空复杂度

时间复杂度：$O(m\log^2 n)$

空间复杂度：$O(m\log^2 n)$

# 72. Sereja and Arcs（SEAARC）

## 72.1 题目大意

有 n 个点，坐标分别是(1,0)，(2,0)，(3,0)，...，(n,0)。每个点有一个颜色，用[1,m]中的整数来表示。

如果两个点颜色相同，那么就过这两个点画一条弧。所有弧都在第一象限。

求有多少对不同颜色的弧相交了。

$1 \leqslant n、m \leqslant 10^5$

## 72.2 题目解法

直接求有多少对弧相交不太方便，可以求有多少对弧，然后减去不相交的。

不相交有两种情况，一种形如“AABB”，另一种形如“ABBA”。“AABB”比较容易求出，考虑求“ABBA”的情况。

如果每种颜色的点数都很小，那么弧的总数的比较小，可以求出所有弧，按右端点的横坐标排序，求出相交的数量。

如果颜色数很少，那么可以枚举两种颜色，计算出由这两种颜色形成的“ABBA”的情况数。

可以选择一个合适的 k，点数不超过 k 的颜色用第一种方法做，它们形成的弧数量不多。点数超过 k 的颜色用第二种方法做，它们的颜色数不会太多。如果两条相交的弧中一条弧属于第一种情况，另一种属于第二种情况，仍然可以用第二种方法做，只不过不是枚举两种颜色，而是枚举点数较多的颜色，点数较少的颜色对应的弧暴力计算。

当 k 取 $\sqrt{\frac{n}{\log n}}$ 附近时，时间复杂度最优。

## 72.3 时空复杂度

时间复杂度：$O\left(n\sqrt{n\log n}\right)$

空间复杂度：$O\left(n\sqrt{\frac{n}{\log n}}\right)$

# 73. Room Corner（ROC）

## 73.1 题目大意

有一个房间，用一个 n×m 的字符阵描述这个房间。“+”、“-”、“|”描述房间的边界，空格填充其他位置。这个房间是连通的，并且如果画一条水平线段，它的端点都在房间内，那么整条线段都在房间内。

房间中所有$90^\circ$的内角处的格子中都站着一个小朋友。相邻的小朋友可以交换位置，交换位置时手必须摸着墙走，相邻就是交换时沿着墙走的过程中不经过属于其他小朋友的$90^\circ$内角。一次交换必须走到对方所在内角处，不允许中途改变行走方向。

移动到一个相邻的格子需要一个时间单位。有 T 次询问，每次询问是，通过合理安排交换方式，两个小朋友相遇最少需要多长时间。相遇是在同一格子或同时向对方所在格子行走。

T≤10000，n、m≤2500

## 73.2 题目解法

找到一个小朋友，沿着墙遍历房间，可以得到一个环。

对于一次询问，两个小朋友一定是相向移动，直到相遇。假设要求 A、B 相遇，当交换到出现“ACB”的情况时，需要 A 等待 B、C 交换，形成“ABC”，然后 A、B 再交换。或者是 B 等待 A、C 交换。可以发现，最优的方案是 A、B 都走到他们的中点两侧，然后再交换，如果有人先走到，就等对方。例如“ACDEFGB”，假设 A、B 的路径中点在 D、E 之间，那么 A 交换到 D 处，B 交换到 E 处，然后再让 A、B 交换。中点所在区域可以通过二分找到。

因为是一个环，所以两个方向都要考虑。

## 73.3 时空复杂度

时间复杂度：$O(nm+T\log(nm))$

空间复杂度：$O(nm)$

# 74. Across the River（RIVPILE）

## 74.1 题目大意

Ciel 想搭一座跨过河的桥。河是一条无限长，宽度为 w 的直线。所有在平面直角坐标系中符合$0 \le y \le w$的点都属于这条河流。

河面上有 n 个木桩，给出木桩的坐标。还有 m 种可以用的木头圆盘。每种圆盘都有半径和价格。

Ciel 可以买任意多的圆盘，放在河面上。每一个圆盘的中心都必须在某一个木桩的位置。

Ciel 只能在地面或圆盘上移动。求搭一座跨过河的桥的最少的花费。

1≤n、m≤250

## 74.2 题目解法

可以考虑最短路。

记河的两侧为 S、T。可以把一个木桩 i 拆成 m 个点(i,j)，对应着可能放的 m 种圆盘，表示在木桩 i 上放圆盘 j 时，从 S 到达这里的花费。若在木桩 i 上放圆盘 j 能直接到达 S 或 T，就在它们之间连边。若木桩 a、b 之间的距离不超过它们上面的圆盘 c、d 的半径和，就在(a,c)和(b,d)之间连边。

图中的边权是到达新木桩时，新木桩上的圆盘的花费。那么，这个图上 S 到 T 的最短路径长度就是最小花费。

但这个图上的边太多了。对于(a,c)，假设它能到达(b,x)、(b,y)、(b,z)，如果 x、y、z 的半径依次递增，就从(a,c)向(b,x)连边。然后从(b,x)向(b,y)连边，边权是圆盘 x 与 y 的花费差，表示付出这些花费使木桩 b 上的圆盘从 x 替换成 y。(b,y)也要向(b,z)连边。这样建图可以减少边的数量。

## 74.3 时空复杂度

时间复杂度：$O(n^2 m \log(nm))$

空间复杂度：$O(n^2 m)$

# 75. A game on a graph（HAMILG）

## 75.1 题目大意

有一个 n 个点 m 条边的无向图，A 和 B 在图上玩游戏。A 先选择一个点作为起始点，然后从 B 开始轮流操作。每次操作是从当前点沿一条边移动到一个没有到达过的点。无法移动的人就输了。

求有多少个点可以作为起始点使 A 必胜。

$1 \leqslant n \leqslant 2000$，$m \leqslant 3 \times 10^5$

## 75.2 题目解法

一个点作为起始点可以必胜，当且仅当存在一个图的最大匹配使得这个点是孤立点。

可以通过带花树求出任一最大匹配。从当前的孤立点出发，长度为偶数的增广路被增广后会产生一个新的孤立点。可以从每一个孤立点出发尝试增广，距离为偶数的点以及花中的点都有可能成为孤立点。

统计可能是孤立点的点数就可以了。

## 75.3 时空复杂度

时间复杂度：$O(nm)$

空间复杂度：$O(m)$

# 76. Future of draughts（CLOWAY）

## 76.1 题目大意

给出 T 张图，第 i 张图有 $n_i$ 个点、$m_i$ 条边。

有 Q 次询问。每次询问给出一个区间 $[l,r]$，再给出一个操作次数 k。对于编号在 $[l,r]$ 中的图，先在每个图中选择一个点作为当前结点，并称之为初始结点。每一次操作，是从这些图中选择至少一个图，将选出的图上的当前结点修改为与当前结点有边相连的另一个点。若一次操作后，编号在 $[l,r]$ 中的图上的当前结点都是它们的初始结点，则不再进行操作。求 k 次操作内停止操作的方案数。答案对 $10^9+7$ 取模。

1≤n、T≤50，1≤k≤10000，Q≤$10^5$

## 76.2 题目解法

可以先预处理出在第 i 张图中长度为 j 的回路数量 $f[i][j]$。这可以通过矩阵乘法求出。$f[i][j]$ 是第 i 张图的邻接矩阵的 j 次方的对角线上数的和。

可以求出邻接矩阵的特征多项式，从而得到 $f[i][j]$ 的线性递推式，可以预处理出所有的 $f[i][j]$。

通过容斥，可以求出对 $[l,r]$ 中的图进行操作，恰好 k 次操作后停止的方案数是 $\sum_{i=1}^{k}\prod_{j=l}^{r} f[j][k]\cdot C_k^i\cdot(-1)^{k-i}$。那么，答案就是这些值的前缀和。

可以枚举 L、R，预处理出所有可能出现的询问的答案。这可以通过 fft 优化。由于要对 $10^9+7$ 取模，所以需要选取三个 ntt 模数分别求值，然后合并出答案。

## 76.3 时空复杂度

时间复杂度：$O\left(Tn^4+Tnk+T^2k\log k\right)$

空间复杂度：$O(Tm+k)$

# 77. A Game of Thrones（GTHRONES）

## 77.1 题目大意

纸上有 n 种数字。给出每种数字是什么，并给出出现次数。

A 和 B 玩游戏。A 先选择一个数字，作为当前数字。然后从 B 开始轮流操作。一次操作是，选择一个数 v，与当前数字 u 相差一个质因子，然后删去 u，v 作为当前数字。无法操作就输了。

相差一个质因子的意思是，若 u<v，那么 u|v，并且 v/u 是一个质数。

求 A 必胜还是 B 必胜。如果 A 必胜，再求出最初选择的数字最小可以是多少。

1≤n≤500

## 77.2 题目解法

每个数字都看做一个点，能相互转移的数字之间连边。游戏就变成了，A 选择起点，从 B 开始轮流从图上走，不允许经过重复的点。

预处理出每两种数字之间是否有边，判断两个数的商是否是质数时，可以使用 miller-rabin 算法。

一个点作为起始点可以必胜，当且仅当存在一个图的最大匹配使得这个点是孤立点。这个图是一个二分图，所以可以通过网络流求最大匹配。枚举每一种数，判断是否能够成为孤立点，可以预先删去一个这种数，然后求最大匹配，判断是否与原图最大匹配相等。

每一种数都做一次最大流太慢了，可以通过退流来优化。

## 77.3 时空复杂度

时间复杂度：$O(n^5)$

空间复杂度：$O(n^2)$

# 78. Little Elephant and Colored Coins（LECOINS）

## 78.1 题目大意

有 n 种硬币，给出每种硬币的价值 v 和颜色 c。每种硬币都有无限个。

m 次询问，每次询问给出一个 Sum，问能否选出一些硬币，使得选出的硬币的价值之和等于 Sum。若可以，求选出的硬币中最多有多少种颜色。

$1 \leqslant n \leqslant 30$，$1 \leqslant v \leqslant 2 \times 10^5$，$1 \leqslant m \leqslant 2 \times 10^5$，$1 \leqslant Sum \leqslant 10^{18}$

## 78.2 题目解法

若求是否存在方案，可以任选一种硬币，记下其价值 w。建 w 个点，表示模 w 意义下的值。对于每一个点 i 和每一种硬币价值 v，从 i 向 $(i+v) \bmod w$ 连一条长度为 v 的边。以 0 为起点，跑最短路，记 0 到 i 的最短路是 $f[i]$。那么，对于询问的 Sum，若 $f[Sum \bmod w]$ 的值不超过 Sum，说明存在一种方案选出价值和为 Sum 的硬币。

由于有颜色，所以可以建 $nw$ 个点。到点(i,j)的最短路 $f[i][j]$ 表示用 i 种颜色的硬币，组成的价值和为 S，且 $S \bmod w = j$，那么 S 最小是 $f[i][j]$。

对于每一种颜色，建边更新最短路。当进行下一种颜色的最短路计算时，删除这一次的边，避免对下一次造成影响。

可以发现，图形成了一个个简单环，所以不必通过最短路算法求值。只需对于每一个环，沿有向边方向绕环更新两圈，就得到最短路了。

## 78.3 时空复杂度

时间复杂度：$O(n^2v+nm)$

空间复杂度：$O(nv)$

# 79. Fibonacci Number（FN）

## 79.1 题目大意

$f[n]=n,n\le 1$

$f[n]=f[n-1]+f[n-2],n>1$

找到最小的非负整数 n，满足 $f[n]=c(\mathrm{mod}\ p)$

共 T 组数据。

1≤T≤100，11≤p≤$2\times10^9$，p 是素数，$(p \bmod 10)$是完全平方数

## 79.2 题目解法

斐波那契数列的通项公式是，

$$f[i]=\frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^i-\left(\frac{1-\sqrt{5}}{2}\right)^i\right)$$

根据 p 的特点，整数存在逆元，$\sqrt{5}$ 存在等价的整数。

记 $x=\frac{1+\sqrt{5}}{2}$，$y=\sqrt{5}\cdot c$

由于 $\frac{1+\sqrt{5}}{2}\cdot\frac{1-\sqrt{5}}{2}=-1$，n 需要满足，

$x^n-(-x)^{-n}=y$

记 $t=x^n$，那么

n 为奇数时，$t+t^{-1}=y$

n 为偶数时，$t-t^{-1}=y$

都能够解出 t 的值。

然后求最小的非负整数 n 满足 $x^n=t$，这可以用大步小步的方法。

奇数情况下求出的值和偶数情况下求出的值中的较小值就是答案。

## 79.3 时空复杂度

时间复杂度：$O(\sqrt{p})$

空间复杂度：$O(\sqrt{p})$

# 80. Payton numbers（CUSTPRIM）

## 80.1 题目大意

定义三元组的乘法，

```
def multiply((a1,b1,c1), (a2,b2,c2)):
  s = (a1a2 + b1b2 + c1c2) + (a1b2 + b1a2) + (c1 + c2)
  t = floor[s/2] + 16(c1 + c2) - c1c2
  A = (t - 2(a1b2 + b1a2) - (a1c2 + c1a2) + 33(a1 + a2)+ (b1b2 - a1a2))
  B = (t - 5(a1b2 + b1a2) - (c1b2 + b1c2) + 33(b1 + b2)+ (2b1b2 + 4a1a2))
  if s is even: return (A-540,B-540,24)
  else: return (A-533,B-533,11)
```

定义 zero：若 x*任何 y=0，则称 x 是 zero。

定义单位元，若 x*任何 y=y，则称 x 是单位元。

定义质数，若 x 不是 zero，且不能被分解成两个非单位元的三元组的乘积，则称 x 是质数。

给定一个三元组，问它是不是质数。

共 T 组数据。

$1 \leqslant T \leqslant 10000$，$-10^7 \leqslant a、b \leqslant 10^7$，$c = 11$ 或 $24$

## 80.2 题目解法

令 w 满足 $w^2 = w - 3$，即 $w = \frac{1+\sqrt{-11}}{2}$。

那么，对于任意一个三元组 $(a,b,c)$，都有到域 $Z[w]$ 的映射：

$$\phi(a,b,c) = (33 - 2a - c) + (b - a)w$$

现在，问题变成了求域 $Z[w]$ 下的一个数 $a + bw$ 是否为质数。

定义共轭，$(a + bw)' = (a + b - bw)$。

令 $Nx = xx'$，可以得到，

- 若 x 不是整数，x 是质数当且仅当 $Nx$ 是质数。
- 若 x 是整数，x 是质数当且仅当满足以下两个条件之一，

  ① $|x| = 2$

  ② $|x| \neq 11$ 且 $-11$ 在模 x 域下没有二次剩余

这些可以通过欧拉判别法和 miller-rabin 来判断。

## 80.3 时空复杂度

时间复杂度：$O(T|a|)$

空间复杂度：$O(1)$

# 81. Team Sigma and Fibonacci（SIGFIB）

## 81.1 题目大意

给出 N、M，求

$$\left(\sum\left(6\cdot x\cdot y\cdot z\cdot fibo[x]\cdot fibo[y]\cdot fibo[z]\right)\right)\%M, x+y+z=N$$

fibo 是斐波那契数列：

$$fibo[0]=0$$
$$fibo[1]=1$$
$$fibo[i]=fibo[i-1]+fibo[i-2], \forall i\geq 2$$

多组数据。

$0\leqslant N\leqslant 10^{18}$，$1\leqslant M\leqslant 10^{5}$，各组数据的 M 的总和 $Sum\_M\leqslant 10^{6}$

## 81.2 题目解法

可以进行如下化简：

$$\begin{aligned}&\sum\left(6\cdot x\cdot y\cdot z\cdot f_x\cdot f_y\cdot f_z\right)\\&=\sum\left(6\cdot x\cdot y\cdot(N-x-y)\cdot f_x\cdot f_y\cdot f_z\right)\\&=N\cdot\sum\left(6\cdot x\cdot y\cdot f_x\cdot f_y\cdot f_z\right)-\sum\left(6\cdot x^2\cdot y\cdot f_x\cdot f_y\cdot f_z\right)-\sum\left(6\cdot x\cdot y^2\cdot f_x\cdot f_y\cdot f_z\right)\\&=N\cdot\sum\left(6\cdot x\cdot y\cdot f_x\cdot f_y\cdot f_z\right)-2\cdot\sum\left(6\cdot x^2\cdot y\cdot f_x\cdot f_y\cdot f_z\right)\end{aligned}$$

其中，

$$\begin{aligned}&\sum\left(6\cdot x\cdot y\cdot f_x\cdot f_y\cdot f_z\right)\\&=\sum\left(\left(2\cdot x\cdot y+2\cdot y\cdot z+2\cdot x\cdot z\right)\cdot f_x\cdot f_y\cdot f_z\right)\\&=\sum\left(\left(N^2-x^2-y^2-z^2\right)\cdot f_x\cdot f_y\cdot f_z\right)\\&=N^2\cdot\sum f_x\cdot f_y\cdot f_z-3\cdot\sum x^2\cdot f_x\cdot f_y\cdot f_z\end{aligned}$$

$$\begin{aligned}&\sum\left(6\cdot x^2\cdot y\cdot f_x\cdot f_y\cdot f_z\right)\\&=\sum\left(\left(3\cdot x^2\cdot y+3\cdot y^2\cdot x\right)\cdot f_x\cdot f_y\cdot f_z\right)\\&=\sum\left(\left(N^3-x^3-y^3-z^3-3\cdot(x+y)\cdot z\cdot(x+y+z)\right)\cdot f_x\cdot f_y\cdot f_z\right)\\&=N^3\cdot\sum f_x\cdot f_y\cdot f_z-3\cdot\sum x^3\cdot f_x\cdot f_y\cdot f_z-N\cdot\sum\left(6\cdot x\cdot y\cdot f_x\cdot f_y\cdot f_z\right)\end{aligned}$$

化简后，只需要求出三个式子的值就可以了，它们是，

$$\sum f_x\cdot f_y\cdot f_z\ 、\ \sum x^2\cdot f_x\cdot f_y\cdot f_z\ 、\ \sum x^3\cdot f_x\cdot f_y\cdot f_z$$

可以发现，$f_x$ 是循环的，循环节长度是 $O(M)$ 的。利用这个性质，可以以

$O(M)$ 的时间复杂度求出这三个式子的值。

## 81.3 时空复杂度

时间复杂度：$O(Sum\_M)$

空间复杂度：$O(Sum\_M)$

# 82. Push the Flow!（PUSHFLOW）

## 82.1 题目大意

给出一个有 n 个点 m 条边的无向连通图 G，每个结点至多属于一个简单环。每条边都有一个容量，支持两种操作：

- 0 S T：求 G 以 S 为源点，T 为汇点时的最大流量
- 1 X NEW_CAPACITY：将第 X 条边的容量修改为 NEW_CAPACITY

$1 \leqslant n \leqslant 10^5$，$0 \leqslant m、q \leqslant 2 \times 10^5$

## 82.2 题目解法

将环看做点，图 G 就变成了树 Tr。一个环是一个双连通分量，可以通过 Tarjan 求出。

最大流等于最小割，可以从最小割考虑。

两个点属于一个环时，最小割是它们之间的两条路径各自的最小边权的和。

两个点不属于同一个环时，需要依次经过一些环，可以在每个环内都求一遍最小割，取最小值。连接两个不同环的边必须经过，也可以只选择它们作为割边，用它们更新一下最小值。

可以把每个环都展开，用线段树维护，这样可以快速得到环内的信息。

在树 Tr 中选择一个点作为根结点，每一个点都对应着图 G 中的一个环，称一个环中离根最近的点为这个环的顶端结点。对于 Tr 中连接 a、b 两点的边(a,b)，令其边权为，以 G 中 a 环、b 环的顶端结点为源汇点时的最大流。这样，计算任意两点 S、T 间的最大流时，除 S、T 所在环的信息要直接计算外，其他信息恰好对应着 Tr 上的一条路径。

可以对 Tr 进行树链剖分，用线段树维护路径信息。

## 82.3 时空复杂度

时间复杂度：$O(n+m+q\log^2 n)$

空间复杂度：$O(n+m)$

# 83. Knight Moving（KNGHTMOV）

## 83.1 题目大意

一张无限大的方格棋盘。初始骑士在(0,0)，骑士有两种移动方式。若当前点是(x,y)，那么可以移动到(x+A,y+B)或(x+C,y+D)。

棋盘上有 n 个障碍格，骑士不能进入障碍格。(0,0)不是障碍格。

给出 A、B、C、D，和障碍格的位置。再给出一个目标点(X,Y)。求骑士有多少种方式到达(X,Y)。

0≤n≤15，输入中涉及到的坐标的绝对值 m≤500

## 83.2 题目解法

若向量(A,B)和(C,D)共线，那么就变成了一维的问题，可以通过 DP 求解。

若向量(A,B)和(C,D)不共线，可以以(A,B)和(C,D)为单位向量，为障碍格和目标点重新计算坐标。在新坐标系中，一次移动可以从(x,y)到达(x+1,y)或(x,y+1)。若不考虑障碍格，从一个点到另一个点的方案数可以用组合数计算。考虑障碍格，可以通过容斥来做。

## 83.3 时空复杂度

时间复杂度：$O(2^n n^2 + m^2)$

空间复杂度：$O(m^2)$

# 84. Cucumber Boy and Cucumber Girl（CUCUMBER）

## 84.1 题目大意

有 m 个披萨大楼，每个大楼中 n 个披萨店。披萨共有 n 种配料，每一个披萨店中都有这 n 种配料。

定义$Q[i,j,k]$，表示第 i 个大楼的第 j 个店中的第 k 种配料的质量。一个披萨的价格是这 n 种配料的质量和。

两个店可以合作，合作生产出的披萨中某种配料的质量是这两个店中这种配料的质量的乘积。

黄瓜男孩和黄瓜女孩要选择两个大楼进行合作。大楼合作的方式就是其中的披萨店两两合作，所以共有$n!$种合作方式。他们一天要吃 n 个披萨，所以他们想让这两个大楼在$n!$天中每天以一种不同的方式进行合作。

如果在某一天中，至少一个披萨的价格是奇数，就会有一个人多付钱。如果这$n!$天中需要有人多付钱的天数是奇数，黄瓜男孩就有机会多付钱的次数比黄瓜女孩多。

黄瓜男孩想知道有多少种选择两个大楼的方式，使得他有机会多付钱次数更多。

1≤n≤60，1≤m≤8000

## 84.2 题目解法

假设现在选择了大楼 s、t。大楼 s 的第 i 个店与大楼 t 的第 j 个店合作生产的披萨的价格是$\sum_{k=1}^{n} Q[s,i,k]\cdot Q[t,j,k]$。

可以用一个矩阵$A_{s,t}$表示 s、t 的合作情况。矩阵中第 i 行第 j 列的数是大楼 s 的第 i 个店与大楼 t 的第 j 个店合作生产的披萨的价格。更简单一些，全部用 0/1 来表示。若价格是奇数，就是 0。若价格是偶数，就是 1。那么，这个矩阵的行列式的值就是需要有人多付钱的天数。在模 2 的意义下，行列式的值若为 0，表示两个大楼不能合作，行列式的值若为 1，表示两个大楼能合作。

若用一个矩阵 S 表示大楼 s 中的信息，第 i 行第 j 列表示第 i 个店中第 j 种配料的价格。同样用矩阵 T 表示大楼 t 中的信息。

可以发现，若令$A=ST^T$，在模 2 的意义下，$A$中的所有数都与$A_{s,t}$中对应位置的数不同。可以在 S 的最后加一列，全部是 1，在$T^T$的最后加一行，全部是 1。那么，$A_{s,t}=ST^T$。

所以，只需要对描述每一个大楼的矩阵（如 S、T），以及它们的转置，求

出行列式，就可以枚举合作的两个大楼，验证是否合法了。

求行列式可以通过高斯消元，由于在模 2 意义下，可以通过二进制压位优化。

## 84.3 时空复杂度

时间复杂度：$O(n^2B+B^2)$

空间复杂度：$O(n^2B)$

# 85. Martial Arts（MARTARTS）

## 85.1 题目大意

有两支队伍，A 队和 B 队，每支队伍都有 n 个队员。A 队与 B 队的队员两两匹配，进行 n 次比赛。一个队的得分是所有比赛中这个队的队员的得分的和。

A 队教练负责安排匹配方式。他想让 A 队的得分减去 B 队的得分尽可能大，在满足这个条件的前提下，再要求 A 队得分尽可能高。

B 队教练和 A 队教练有相同的想法，他想让 B 队的得分减去 A 队的得分尽可能大，在满足这个条件的前提下，再要求 B 队得分尽可能高。

B 队教练会在 A 队教练安排好匹配方式后，取消其中的一次比赛。

给出 A 队的第 i 个队员和 B 队的第 j 个队员进行比赛的结果 f[i][j]，求 A 队教练怎样安排最好，输出此时两队的得分。

1≤n≤100

## 85.2 题目解法

可以枚举取消的那一场比赛是什么。然后把对于 B 队教练来说，比取消的那场比赛结果更好的比赛加入二分图中，对剩下的 n-1 对队员进行最大带权匹配。这可以用 KM 算法。

如果将边排序，问题可以变成，每次先向图中加入一条边，然后求最大权匹配是什么。

不必每次加入一条边后都重新进行一次 KM 算法。可以先将所有边都加入，边权是正无穷。加入一条边的操作，相当于将一条边的边权减小为原值。这样，每次修改操作只需在 KM 算法中再修改较少次数的点上的权值就可以了。

## 85.3 时空复杂度

时间复杂度：$O(n^4)$

空间复杂度：$O(n^2)$

# 86. Misinterpretation 2（MISINT2）

## 86.1 题目大意

对于一个长度为 n 的由小写字母组成的字符串 S，先从左到右将 S 的偶数位的字符写到开头，再从左到右将奇数位的字符继续写下去，得到新字符串 T。

求有多少长度在 L 和 R 之间的 S，满足 S 与 T 相同。

共 T 组数据。

$1 \leqslant T \leqslant 5$，$1 \leqslant L \leqslant R \leqslant 10^{10}$，$R-L \leqslant 50000$

## 86.2 题目解法

R-L 较小，可以对于每一个在 L、R 之间的长度 n 单独计算。

若 n 是奇数，那么 S 中的第 i 个字符对应到 T 中就在第 $2i \bmod n$ 个位置。特殊地，$i = n$ 时，$2i \bmod n = 0$，可以认为第 0 个位置就是第 n 个位置。

若 n 是偶数，那么 S 中的第 i 个字符对应到 T 中就在第 $2i \bmod (n+1)$ 个位置。这相当于，在 S 后面补上一个字符，使 S 的长度为奇数，这时将 S 变换成 T，补上的字符仍在最后，再删去那个字符就可以了。

所以，偶数的情况可以作为奇数的情况来做，那么只需考虑奇数的情况。

S 到 T 的对应关系相当于是一个置换，写成轮换的形式，那么在同一轮换中的位置字符必须相同，不在同一轮换中的位置互不影响。记轮换数为 $f(n)$，那么方案数就是 $26^{f(n)}$。

可以发现，同一轮换中所有位置 i 的 $\gcd(i,n)$ 相同，并且 $\gcd(i,n)$ 相同的 i 所在的轮换长度都相同。所以，可以根据 $\gcd(i,n)$ 分别计算。

记 $ord(x)$ 为 2 模 x 的阶，那么，

$$f(n) = \sum_{d|n} \frac{\varphi(d)}{ord(d)}$$

对 n 分解质因数，然后枚举它的约数进行计算。

可以同时对[L,R]中的所有数同时分解质因数，枚举$[1, 10^5]$中的数 i，可以直接得出哪些数有质因子 i。进行完这一步后，若一个数的质因子还没有全部找出，那么就只可能再有一个质因子，那个质因子可以用这个数试除已经求出的质因子来求出。

约数 d 的 $\varphi(d)$ 比较容易求出，考虑求 $ord(d)$。

可以发现，若 a、b 互质，那么 $ord(ab)=lcm(ord(a),ord(b))$，所以可以对每一个质因子单独求值。可以预处理出$[1,10^5]$中所有质数的$ord(x)$。那么对于每一个 n，至多再计算一个新的质因子。

计算$ord(x)$时，可以对$x-1$的每一个质因子试除进行判断。

## 86.3 时空复杂度

时间复杂度：$O\left(\frac{R^{\frac{3}{4}}}{\log R}+\sqrt{R}\log R+T(R-L)\left(\frac{\sqrt{R}}{\log R}+\log^2 R\right)\right)$

空间复杂度：$O(\sqrt{R})$

# 87. String Query（STRQUERY）

## 87.1 题目大意

给出一个长度为 10 的字符串 S，支持几种操作，

- 在开头插入一个字符
- 在正中间插入一个字符
- 在最后插入一个字符
- 删除第一个字符
- 删除正中间的字符
- 删除最后一个字符
- 给出一个字符串 Q，求 Q 在 S 中出现了多少次

共 m 次操作。

$1 \leqslant m \leqslant 1.5 \times 10^5$，$\sum |Q| \leq 1.5 \times 10^6$

## 87.2 题目解法

若只在结尾插入删除，可以通过后缀平衡树维护。

若只在开头和结尾插入删除，可以维护两棵后缀平衡树 L、R，开头的操作在 L 中进行，结尾的操作在 R 中进行。当其中一棵后缀平衡树中不存在字符了，就暴力重建 L、R。

若需要在正中间插入删除，可以将 L、R 作为一个整体 T，维护两个 T 结构 TL、TR。修改时在 TL、TR 之间交换一些字符，使中间位置在 TR 的头部。

查询时，在同一后缀平衡树中的情况通过后缀平衡树求出，其他情况可以暴力求出。

## 87.3 时空复杂度

时间复杂度：$O\left(m \log m + \sum |Q| \cdot \log m\right)$

空间复杂度：$O(m)$

# 88. Dynamic Trees and Queries（ANUDTQ）

## 88.1 题目大意

给出一棵 n 个点的有根树，点有点权。支持四种操作：

- 给一个点增加一个子结点，给出子结点的点权。
- 删除一个子树
- 给一个子树中所有点的点权都增加一个值
- 求一个子树中所有点的点权和

共 m 次操作。

$1 \leqslant n、m \leqslant 10^5$

## 88.2 题目解法

因为有加入点、删除点的操作，考虑用 LCT 来做。

LCT 不便于处理子树增加一个值的操作。记以 x 为根的子树中共有 $tot[x]$ 个结点，那么当以 x 为根的子树都增加 y 时，x 以及 x 的祖先为根的子树中点权和都增加了 $tot[x] \cdot y$，x 的子树中的点 z 为根的子树中点权和增加了 $tot[z] \cdot y$。

可以对这两种情况分别维护。x 以及 x 的祖先的信息，是一条链上都增加一个值，可以用 LCT 直接维护。对于 x 的子树中点 z 的修改，可以在 x 处标记，所有 x 的子树中的点 z 被查询到时，值要增加 $tot[z] \cdot y$。

那么，当查询一个点时，不仅要考虑这个点上的信息，还要考虑它的祖先上的标记。一个点的祖先结点构成一条链，可以用 LCT 维护。

## 88.3 时空复杂度

时间复杂度：$O(n + m \log n)$

空间复杂度：$O(n + m)$

# 89. Flight Distance（FLYDIST）

## 89.1 题目大意

有一张 n 个点 m 条边的无向图。边长都是不超过 t 的正整数。

可以修改一些边的长度，代价是修改前后边长差的绝对值。修改量是有理数。修改后边长必须仍为正数。

通过修改一些边的长度，使得图满足，对于任意一条连接 a、b 的长度为 c 的边，a、b 间的最短路长度为 c。

求最小代价，用分数表示。

图无重边、自环。

1≤n≤10，1≤t≤20

## 89.2 题目解法

设第 i 条边的原边长 $c[i]$，增加值 $d^+[i]$，减少 $d^-[i]$。设修改后 i、j 之间的最短路为 $dis[i][j]$。

若第 s 条边连接 a、b，那么需要满足，

$$dis[a][i] \le c[s] + d^+[s] - d^-[s] + dis[b][i]$$

由于修改后边长为正，所以，

$$c[i] + d^+[i] - d^-[i] > 0$$

若第 i 条边连接 a、b，那么，

$$dis[a][b] = c[i] + d^+[i] - d^-[i]$$

需要最小化的是，

$$\sum_{i=1}^{m} d^+[i] + d^-[i]$$

这可以通过线性规划求解。

## 89.3 时空复杂度

时间复杂度：$O\left(2^{n^2}\right)$

空间复杂度：$O\left(n^4\right)$

# 90. Substrings on a Tree（TSUBSTR）

## 90.1 题目大意

给出一个 n 个点的有根树，每个点上有一个小写字母。从一个点开始，向后代的方向走，能够得到一条路径，路径上的点上的字母依次写下来，能够得到一个字符串。

求能够得到多少不同的字符串。

还有 m 次询问，每次询问给出一个由 26 个小写字母组成的排列，表示小写字母之间的大小关系。还给出一个数字 x，求能够得到的字符串中，按给出的字母的大小关系来比较，字典序第 x 小的字符串。

$1 \leqslant n \leqslant 2.5 \times 10^5$，$1 \leqslant m \leqslant 50000$，输出字符数不超过 $t = 8 \times 10^5$

## 90.2 题目解法

根据树上的信息，可以建立后缀自动机。

对后缀自动机进行拓扑排序，按拓扑序 dp 求出从每个结点出发，能够匹配多少字符串 。起点出发能够匹配的字符串数量，就是从这个树上能够得到的字符串的数量。

对于每次询问，从起点出发找出那个字符串。在一个点时，按给定的字母顺序枚举，根据能够匹配的字符串数量，验证下一个字母是否是当前枚举到的字母。

## 90.3 时空复杂度

时间复杂度：$O(n+m+t)$

空间复杂度：$O(n)$

# 91. Union on Tree（BTREE）

## 91.1 题目大意

给出一棵 n 个点的树，树边长度都是 1。

有 m 次询问。每次询问先给出一个数 k，接下来给出 k 个条件(a,r)，表示与 a 点距离不超过 r。求有多少点满足这 k 个条件中至少一个。

1≤n、m≤50000，所有询问的 k 的和 $sum \le 5 \times 10^5$

## 91.2 题目解法

若 $k=1$，可以通过点分治求解。

进行点分治时，若当前选出的中心是 x，那么记录这个块中所有点到 x 的距离，以及到 x 距离为 i 的点有多少。并且这些信息对删除 x 后形成的每一个连通块还要单独记录。

求与 x 的距离不超过 y 的点数时，先找到以 x 为中心的块，统计块内的答案。再找出上一层分治中 x 所属块的中心 z，统计在这个块中，但不在以 x 为中心的块中的合法点数。从 x 到达这些点，一定经过 z，若用 dis(x,z)表示 x 与 z 的距离，那么相当于统计与 z 距离不超过 y-dis(x,z)的点数。不断向上一层枚举，统计所有层的答案。

若 $k>1$，可以对给定的关键点建虚树。若有两个条件是(a,b)和(c,d)，那么 c 给 a 的贡献是 $d-dis(a,c)$。若 $b \le d-dis(a,c)$，那么相当于条件(a,b)不存在。否则，会同时被这两个条件记录的点相当于满足另一个条件(e,f)。

可以通过类似最短路的方法求出虚树上每一个点的实际影响范围，统计答案。

## 91.3 时空复杂度

时间复杂度：$O(n\log n + sum\log n)$

空间复杂度：$O(n\log n)$

# 92. Chef and Balanced Strings（CBAL）

## 92.1 题目大意

一个字符串是平衡字符串当且仅当任意一个字符都出现了偶数次。

给出一个长度为 n 的字符串 A，用 A[s,t]表示 A 的第 s 个字符到第 t 个字符组成的子串，用|S|表示字符串 S 的长度。

给出 m 次形如(l,r,type)的询问，求，

$$\sum_{\substack{l\le a\le b\le r\\ A[l,r].is.balanced}} |A[l,r]|^{type}$$

强制在线。

1≤n、m≤$10^5$，0≤type≤2

## 92.2 题目解法

对每一个字符分配一个不同的 2 的整数次幂。一个字符串是平衡字符串当且仅当各个字符对应数字的异或值为 0。对 A 中的字符求出前缀异或值，A 的子串内部的异或值是两个前缀的异或值。问题变成了，区间内部每一对相同的数字对答案的贡献是距离的 type 次幂。

已知[l,r]的答案时，计算[l,r+1]，可以用[l,r]的值加上与 r+1 位置有关的贡献，也就是求出[l,r]中与 r+1 位置的数相等的数的信息。这可以通过前缀和求出。

对 A 进行分块，预处理出所有仅包含完整的块的区间的信息。对于一次询问[l,r]，通过预处理的信息得到完整块对应区间的答案，再将区间左右端点一个个位置扩展得到当前询问的答案。

## 92.3 时空复杂度

时间复杂度：$O(n\sqrt{n})$

空间复杂度：$O(n\sqrt{n})$

# 第二部分　challenge 题目

共 11 题

# 93. The Great Plain（LAND）

## 93.1 题目大意

有一个 n×m 的表格，每个格子上需要填一个正整数。有些格子已经填好了，没有填的格子需要你来填。

令两个格子相邻，当且仅当它们有公共边。对于每一对相邻的格子，若它们填的数的差是 k，那么对整个表格的权值的贡献是 $2^k$ 。

使表格的权值尽可能小。

1≤n、m≤50

## 93.2 题目解法

对于每一个需要填的格子，可以通过某种方式设置一个初始值，然后再尝试修改得更优。

我选择的计算格子 x 的初始值的方式是，对于每一个已经填好的格子 y，它的权值是 x 与 y 的曼哈顿距离的平方的倒数，x 的初始值是所有 y 的加权平均数四舍五入到整数后的结果。

然后枚举每一个格子，计算是否存在另一个整数 t，使得在这个格子填 t 时，整个表格的权值更小。若存在，则修改。

可以多次枚举所有格子，多次验证是否修改。

## 93.3 时空复杂度

设枚举格子验证是否修改共 tot 次。

时间复杂度：$O(n^2m^2+tot\cdot nm)$

空间复杂度：$O(nm)$

# 94. Stepping Average（STEPAVG）

## 94.1 题目大意

有 n 个数，每次选择两个数，删除它们，再将它们的平均数加入，直到最后只剩一个数。

找一种方案，使最后剩下的那个数尽可能接近 m。

$n=1000$，数据随机

## 94.2 题目解法

由于操作方式特别多，可以只考虑特定的操作方式。

我选择的操作方式是，先选择两个数，删除它们，加入它们的平均数 x。然后再选择 x 和另一个数，删除它们，加入它们的平均数 y。然后再选择 y 和另一个数……

这样相当于，对这 n 个数任意排序，得到一个长度为 n 的数列 A。计算 A[1]、A[2]的平均数 x，在计算 x 与 A[3]的平均数 y……

这样，最终得到的数 Ans 是，

$$Ans=\frac{A[1]}{2^{n-1}}+\frac{A[2]}{2^{n-1}}+\frac{A[3]}{2^{n-2}}+\frac{A[4]}{2^{n-3}}+\frac{A[5]}{2^{n-4}}+\ldots+\frac{A[n-1]}{2^{2}}+\frac{A[n]}{2}$$

考虑从后往前进行选择，即先选择 A[n]。若选择 x 作为 A[n]，那么前 n-1 个数的选择相当于是一个递归子问题，是对那 n-1 个数进行排序，使它们计算出的 Ans 尽可能接近 $\frac{x+m}{2}$。

选择 x 时，可以使其余的 n-1 个数的平均数尽可能接近 $\frac{x+m}{2}$。选择出合适的 x 后，递归下去继续选择。

## 94.3 时空复杂度

时间复杂度：$O(n^2)$

空间复杂度：$O(n)$

# 95. Closest Points（CLOSEST）

## 95.1 题目大意

三维空间中有 n 个点，有 m 次询问，每次询问给出一个点 P，求与 P 最近的是哪一个点。

使正确的回答次数尽可能多。

$n = m = 50000$

## 95.2 题目解法

设三个坐标分别为 x、y、z。与 P 最近的点的三个坐标都会和 P 相差不大。

可以将点按 x 排序，二分出 P 在排序后的点列中的位置，从这个位置向左向右各枚举一定数量的点，计算它们与 P 的距离。

同样的方法，也对按 y 排序、按 z 排序的点列进行枚举。

## 95.3 时空复杂度

设向左向右枚举的点的数量为 t。

时间复杂度：$O(n\log n + tm)$

空间复杂度：$O(n)$

# 96. Simultaneous Nim（SIMNIM）

## 96.1 题目大意

有 n 个数，将它们分组，要求每一组中数的异或值都为 0。找一种分组方案，使组数尽可能多。

保证所有数的异或值为 0。

每一个数都是$[1,2^m-1]$中的整数。

10≤n≤1000，5≤m≤60

## 96.2 题目解法

对这 n 个数求出一组线性基，不在线性基中的数 x 可以表示成线性基中的一些数的异或值，那么可以令 x 与线性基中的这些数分成一组。

枚举所有可能的 x，选择使这一组中的数尽可能少的 x，将这些数分成一组。

然后对剩下的数继续求线性基、分组，直到所有数都被分组。

## 96.3 时空复杂度

时间复杂度：$O(n^2m)$

空间复杂度：$O(n)$

# 97. Maximum Sub-rectangle in Matrix（MAXRECT）

## 97.1 题目大意

有一个 n×m 的矩阵，选中一些行，再选中一些列。对于矩阵中的一个数，若它所在行和所在列都被选中了，那么它就被选中了。

使被选中的数的和尽可能大。

200≤n、m≤300

## 97.2 题目解法

随机一些行，选择这些行，然后枚举每一列，计算这一列中被选择的行的数的和是否大于 0，若大于 0，那么就选择这一列。这样就确定了选择哪些列。然后根据选择的列重新确定选择哪些行，再根据这些行确定选择哪些列，重复执行 t 次。得到最终的选择。

可以多次执行上述操作，每次可以随机不同的初始行集合。取这 s 次执行操作中的最优值。

## 97.3 时空复杂度

时间复杂度：$O(stnm)$

空间复杂度：$O(nm)$

# 98. To challenge or not（CHAORNOT）

## 98.1 题目大意

给出 n 个数，从中选出尽可能多的数，使选出的数中，不存在三个数构成等差数列。

数据生成时，先从$[10^4,10^5]$中随机一个整数 L，再从$[0.1,0.9]$中随机一个实数 p，然后$[0,L)$中的每一个整数都有 p 的概率出现在输入中。

## 98.2 题目解法

将数从小到大进行排序，然后从小到大枚举每一个数，求选出后是否会构成等差数列，若不会，就选出这个数。

当选出一个数 x 时，枚举所有之前已经被选出的数 y，标记$2y-x$不能被选择。这样，每当枚举到一个数时，就可以直接判断它能否被选出了。

## 98.3 时空复杂度

设选出的数的数量为 m。

时间复杂度：$O(n\log n+m^2)$

空间复杂度：$O(n)$

# 99. Sereja and Sorting（SEASOR）

## 99.1 题目大意

给出一个长度为 n 的数列，你要将它们从小到大排序。

一次操作可以选择一个长度为 m 的区间，将这个区间中的数进行排序。

找出一种方案，使操作数尽可能少。

2≤m≤n≤1000

## 99.2 题目解法

用冒泡排序的过程，可以想到一种排序方式，就是每次将最小的数排到最前面，然后再将第二小的数排到第二……

将一个数 x 排到特定位置 y 时，可以先找到 x 当前位置 z，对$[z-m+1,z]$进行排序，这样 x 就到了 z-m+1 的位置，不断进行下去，直到 x 到了 y 位置。

这样的操作数较多，因为一次只能将一个数排序到正确位置。

考虑同时对多个数进行排序。设一次同时对 t 个数进行排序。若这些数中最靠右的数的位置是 z，最小的数需要排到的位置是 y。那么，先对$[z-m+1,z]$进行排序，再对$[z-2m+t+1,z-m+t]$进行排序，直到这 t 个数都到了正确位置。

一次对 t 个数进行排序，要进行$\frac{n}{t}$次。一次中，要对$\frac{n}{m-t}$个区间进行排序。所以，总操作数约为$\frac{n^2}{t(m-t)}$次。

当 t 取$\frac{m}{2}$左右时总操作数较少。

## 99.3 时空复杂度

时间复杂度：$O\left(\frac{n^2\log m}{m}\right)$

空间复杂度：$O\left(\frac{n^2}{m^2}\right)$

# 100. Deleting numbers（DELNMS）

## 100.1 题目大意

给出一个长度为 n 的数组 a[n]。需要通过进行一些操作，删除所有的数。

每次操作可以选择 v，t 两个数，满足 $v+t\leq n+1$，且令 k 为最大的满足 $v+kt\leq n$ 的数，必须满足 $a[v]=a[v+t]=a[v+2t]=...=a[v+kt]$。然后这些数将会被删除。之后形成新的由剩下的 n-k 个数组成的数组 a[n-k]。

寻找一种操作方案，使操作数尽可能少。

$1\leqslant n\leqslant 10^5$，数组中的数都是正整数且不超过 $m\leqslant 10^5$

## 100.2 题目解法

有一种方案就是一次操作只删除一个数。

稍微优化一下，找出出现次数最多的那个数 x，把不是 x 的数一个个删除，然后剩下的所有 x 都可以在一次操作中删除。

可以发现，对于两个数 x、y，若最靠右的 x 在最靠左的 y 的左边，那么 x、y 各自可以用一次操作都删除。它们互相不影响。

这样，对于一个值 x，用它出现的最靠左和最靠右的位置，表示出它的覆盖区间。这个区间的权值就是这个值的出现次数。现在需要找出一些互不相交的区间，最大化权值和。这可以通过排序 dp 求出最大权值和与操作方案。

## 100.3 时空复杂度

时间复杂度：$O(n\log n)$ -

空间复杂度：$O(n)$

# 101. Chef and Painting（CHEFPNT）

## 101.1　题目大意

有一个 n×m 的网格，有些格子是白色的，有些格子是黑色的。

一次操作的步骤是，

- 选择一个白色格子 P
- 选择方向，上下或左右
- 若选择的是上下，就从 P 开始向上，把经过的格子涂成黑色，直到碰到之前就是黑色的格子或到了网格边界。然后再向下进行同样的操作。若选择的是左右，就向左向右进行这样的操作。

现在需要将所有格子都涂成黑色，寻找一种方案，使操作数尽可能少。

1≤n、m≤100

## 101.2　题目解法

最简单的想法就是所有操作都选择相同的方向。可以计算出全部选择上下方向和全部选择左右方向时的方案和操作数。

也可以加一个修改。每当选择一个白色格子，计算选择上下方向和左右方向时能够涂黑的格子数。选择能够涂黑的格子数多的方向。但这样不一定比原来更优。

把这些想法都实现一遍，选择操作数最少的方法。

## 101.3　时空复杂度

时间复杂度：$O(nm(n+m))$

空间复杂度：$O(nm)$

# 102. Kali and Devtas（KALKI）

## 102.1 题目大意

给出平面上的 n 个点，求一种生成树方案，使得 C[i]的最大值尽可能小。

C[i]的计算方式是，对于每个点 P，设在生成树中与 P 有边的点中，离 P 最远的点为 Q，P、Q 的距离为 R，那么所有与 P 的距离不超过 R 的点的 C[i]值都加一（包括 P 本身）。

数据随机。

3≤n≤400

## 102.2 题目解法

对于一个点 P，它连出去的边的长度一定是越小越好。

由于 C[i]的计算方式较复杂，而且数据随机，所以可以直接求最小生成树。

最小生成树的 C[i]的最大值，与最优解的 C[i]的最大值差距不会太大。

## 102.3 时空复杂度

时间复杂度：$O(n^2 \cdot \log n^2)$

空间复杂度：$O(n^2)$

# 103. Sereja and Shuffling（SEASHUF）

## 103.1 题目大意

给出一个长度为 n 的数列，n 是偶数。

可以进行一些操作，一次操作是，选择一个区间[l,r]，然后将数列中的第 l 个数移动到第 r 个数的后面，代价是区间长度，即 r-l+1。

所有操作的代价和不能超过 2n。

将最终的数列从中间分成等长的两部分，设前$\frac{n}{2}$个数的和为 a，后$\frac{n}{2}$个数的和为 b，令$S=|a-b|$。

寻找一种操作方案，使 S 尽可能小。

数据随机。

$2\leqslant n\leqslant 10^5$，数列中的数 m 满足 $0\leqslant m\leqslant 10^9$

## 103.2 题目解法

由于数据随机，所以当 n 很大时，可以只考虑对中间位置附近的数进行移动。

为了方便，先不考虑操作具体是什么，先考虑每个数最终属于哪半部分，然后可以通过贪心的方法得到操作方案。

当$S\geqslant 2\times 10^9$时，从某半部分中选择一个尽可能小的数，另半部分中选择一个尽可能大的数，改变它们所属的部分。由于不同选择方式的代价不同，可以将 S 的改变量和代价结合起来，构造一个估价函数，选择数进行改变。

当$S<2\times 10^9$时，也可以用这个方法来修改，不过要考虑绝对值的问题，所以不再是修改量越大越好。

这样得到的方案就已经很优了。最后可以随机选择几个数，重新分配它们所属的部分，搜索所有情况，取最优方案。

## 103.3 时空复杂度

若 n 很大时，只考虑中间位置附近的 m 个数。

时间复杂度：$O(nm)$

空间复杂度：$O(n)$