数学

86D

- 莫队，开个桶记录当前每种数字出现个数。
- 移动端点的时候数字增加/减少，莫队算一下差值即可。

484B

- 开个桶记录当前每种数字是否出现。
- 从小到大枚举出现过的数作为除数 i，再枚举商不超过 x（i*x<=1e6），那么被除数<i*(x+1)，求一下前缀最大值 y，用 y-i*x 更新答案。

474F

- 得**满分的必须是区间所有数的约数**。那么这个数首先得是区间最小值。
- 求出区间 gcd，区间最小值以及最小值出现次数。
- 如果最小值不整除 gcd，那么没人满分，否则满分的就是最小值出现次数。
- 这个可以离线，开个栈维护不同的后缀 gcd，后缀最小值和出现次数，在右端点求答案即可。

1450C1（#）

- 考虑 X 是所有点的做法。
- 如果所有点都是 X，显然有 3 种简单的染 O 方式且两种染法间没有重复点，即染(i+j)%3=0/1/2。
- 那么只有一部分点是 X 的方案相当于 X 的位置和它们求交，必定有一个不超过 k/3。

438D

- **一个数如果模一个比它小的数，那么模 log 次就变成 0 了**，原因是每次至少减半。
- 那么直接线段树记录区间最大值，如果最大值不超过模数就递归。
- 因为区间最大值不到了就会退出，而每个点到根的链长度只有 log n，所以总复杂度是 log^2。

559C

- **经典 k^2 容斥**，考虑用总方案数减去经过黑点的方案数。
- 对于一条有黑点的路径，在第一个黑点的时候把它算上。
- 设 dp[i]表示走到点 i，且经过的第一个黑点就是 i（即之前没有黑点）的方案数。
- 转移也考虑用总方案数减去之前经过黑点的方案数，枚举路径经过的第一个黑点，dp[i]=C(xi-1+yi-1,xi-1)-\sum dp[j]+C(xi-xj,yi-yj,xi-xj)。
- 方便起见可以将右下角的点也当作黑点，直接输出它的 dp 值即可。

1485D（#）

- **lcm=2^4*3^2*5*7*11*13=720720 不超过值域限制。**如果 k 可以为 0，那么所有数都填 lcm 即可。
- 黑白染色，所有限制都为黑-白，给黑点加上 v[i][j]^4

1499D

- 推式子，lcm 一定是 gcd 的倍数，把式子改写为 c*(lcm/gcd)=x/gcd+d，lcm=(x+d*gcd)/c。
- 那么 gcd 一定是 x 的约数，枚举 gcd，可以解出 lcm。
- 由于 gcd 是幂次上取 min，lcm 是幂次上取 max，如果幂次上 min 和 max 不同则有两种方案，否则只有一种。
- 那么 lcm/gcd 就是幂次上 max 和 min 之差，非 0 的就是 min 和 max 不同，方案数就是 lcm/gcd 的质因子个数。
- 直接预处理即可。

1355E

- 怎么说呢，关键在于你发现**不可能三种操作都做**，做 AR 和做 M 等价，哪个小做哪个。
- 如果只用 A,R，假设最后的值为 x，那么<x 的用 A，>x 的用 B，你直接暴力枚举 x 显然不太行，不过 x 不可能是 h 之外的数，否则如果<x 的更多你可以增加 x，>x 的更多你可以减小 x。
- 如果只用 A,M，假设最后的值为 x，则要用多少次 A 是固定的，为 nx 减去 h 之和，M 操作要使 h 中的数都<=x，用 M 的次数其实也是固定的，就是 max(h_i-x,0)之和。x 不可能是 h 之外的数，否则可以减小/增大。
- 如果只用 R,M，假设最后的值为 x，则要用多少次 R 是固定的，为 h 之和减去 nx，M 操作要使 h 中的数都>=x，用 M 的次数其实也是固定的，就是 max(x-h_i,0)之和。x 不可能是 h 之外的数，否则可以减小/增大。
- 三种情况分别计算即可。

1389D

- 不妨假设 l1<=l2（如果 l1=l2 则不妨假设 r1<=r2），先判断掉交*n>=k 的情况，直接输出 0.
- 考虑 l1<=r1<=l2<=r2 的情况，那么先需要 l2-r1 次变成 r1=l2，然后需要 r2-l1 次变成 l1=l2,r1=r2，且过程中每次交+1，最后每两次操作交+1。
- 考虑 l1<=l2<=r1<=r2 的情况，那么需要 l2-l1+r2-r1 次变成 l1=l2,r1=r2，且过程中每次交+1，最后每两次操作交+1。
- 考虑 l1<=l2<=r2<=r1 的情况，那么需要 l2-l1+r1-r2 次变成 l1=l2,r1=r2，且过程中每次交+1，最后每两次操作交+1。
- 判断一下，你首先一定先把一个变成 l1=l2,r1=r2。
- 一种选择是直接做每两次操作交+1。
- 一种选择是尽量先做变成 l1=l2,r1=r2，如果剩下的次数不足以做，再考虑剩下的用每两次操作交+1 还是尽量做 l1=l2,r1=r2；如果把 n 条都做完了，那只能做每两次交+1。

1493D

- 用 multiset 存每个质数的出现位置（出现多次则存多次）。
- 更新的时候分解质因数，对当前质因数的 map 的 i 位置+1，记录一下 map 里没出现过的数字个数，如果为 0，就暴力清一下，给答案乘上当前质因数。
- 每次加入最多 log 个质因子，每个数最多被删去一次。

1538G

- 先交换使得 x<=y,a<=b。你可以认为是**取(a+b,a+b)（贡献为 2）,(a,b)（贡献为 1）**。
- 你先取(a,b)，直到再取就 x>y 了，然后你取(a+b,a+b)；或者取到正好 x>y，然后你取(a+b,a+b)。
- 最后如果还有多余就取(a,b)。你到了正好 x>y 后取两个(a,b)即(a+a,b+b)一定不如取(a+b,a+b)。

1333F

- 从 n 个数都有开始删。你考虑你会怎么删：
  - 如果 a,b 都在 S 中且 a|b，那么 b 一定比 a 先删（b 和其它数的 gcd 一定不比 a 和其它数的 gcd 要小）；
  - 那如果一个数保留那么它的约数都会保留，所以任意两数的 gcd 一定在集合中。
  - 对于集合内的数，它的最大约数（非自己）就是它和比它小的数能组成的最大的 gcd。
- 那么集合的权值就是所有数最大约数的最大值，你每次一定删除最大约数为最大值的数。
- 按照最大约数的大小排序即可，按顺序输出。

1188B

- 这个**(a+b)(a^2+b^2)=x** 属于经典式子了，两边同乘(a-b)就可以合并两次平方差，变成 **a^4-b^4=x(a-b)**。
- 对于本题，由于 p 是质数且 a 两两不同，所以也可以使用这个式子，变成 a_i^4-b_i^4≡k(a_i-b_i) (mod p)，移项变成 a_i^4-ka_i≡b_i^4-kb_i (mod p)，map 记一下每个的个数即可。

1438D（#）

- 操作后**异或和不变**。
- 若 n 是奇数，设异或和为 x，只能把所有数都变成 x。操作(1,2,3),(1,4,5),(1,6,7)…，可以位置 1 变成 x，且 2=3,4=5…，接下来再做一次所有操作即可。
- 若 n 是偶数，最终异或和为 0，先判断无解，否则删掉最后一个做 n 为奇数即可。

1363D

- 这题的关键在于你要**考虑原序列的最大值**，如果最大值在第 x 个子集内，则其它子集的 P 都为最大值，再询问一次即可得到 x 子集的值；如果最大值不在子集内，那么所有子集的 P 都为最大值。
- 直接一次询问出 mx，最大值的位置可以直接二分，看[l,mid]的最大值，是 mx 就到 r=mid，否则 l=mid+1。
- 最多使用 12 次操作。

1474D

- 如果没有交换两个数，你可以直接从左到右模拟或从右到左模拟。
- 有交换相邻两个数，你求出左边从左到右留下的数，右边从右到左留下的数，然后中间四个暴力判一下即可。

486D

- 为了防止出现重复，定义两个点大小关系为(a_i,i)的大小关系。

- 因为 n,d<=2000，考虑枚举每个连通块最大的点，然后树形 dp，要求不能经过比它大的或点权之差>d 的。
- 转移就是 dp[u]为所有 dp[v]的乘积+1，最后算答案时要-1（去掉整个是空的）。

1110E（#）

- **差分，变成交换相邻两个 d**，排序后判断两个是否相等即可。

1332E（#）

- 由于你可以给每个格子不停的+2，所以可以将初始的 a 都 mod 2，之后操作都**在 mod 2 意义下**。
- 可以把操作从将相邻两个数同时+1 改成任意两个数同时+1（找一条这两个数间的路径，相邻都做一次操作）。
- 记 tot=H^{nm}。
- 若 n*m 为奇数，那么 0/1 一定有一种个数是偶数，操作即可。(tot)
- 若 n*m 为偶数，如果是 0/1 个数都是偶数则也可操作，都是奇数则无法操作。(tot/2)

903D

- 不妨先当作 d(x,y)=y-x 算，对于|x-y|=1 的最后再减去贡献。
- 那么 d(x,y)之和就是 a[i]*i-a[1..i]前缀和，再加上每个 i，之前为 a[i]+1 的数字个数-之前为 a[i]-1 的数字个数。

1322B（#）

- 考虑依次确定每一位的值，两个数加起来第w位是 1 当且仅当两个数的后k位之和满足 $2^k \le a_i + a_j < 2^{k+1}$或$2^{k+1} + 2^k \le a_i + a_j$。双指针算一下看奇偶性即可。到下一位的时候**基数排序更新顺序**。

1467D

- 具体是哪些路径不太好算，但是对于**某个点一共经过了几次**，还是可以算的。
- 设 dp[i][j]表示从 i 出发走 j 步的方案数，转移就是 dp[i][j]=dp[i-1][j-1]+dp[i+1][j-1]。
- 那么经过的总次数可以枚举它在第 x(0<=x<=k)步出现，方案数是 dp[i][x]*dp[i][k-x]，加起来即可。
- 求答案的时候，减去原来第 i 个点的贡献，修改权值，加上第 i 个点的贡献即可。

1428E

- 经典**堆贪心**。每次多切一次胡萝卜，新增的贡献是减小的。问题相当于有 n 个递减序列取出最大的 k 个数。
- 那么拿个大根堆堆来维护，权值为再切一次能新增的贡献，每次取堆顶弹出即可。

1117D

- 直接组合数乱算好像比较复杂，也不太好实现。它这个 m 很小好像并没有用到。
- 考虑求个递推式，在最后放的魔法宝石要不要分解。f_n=f_{n-1}+f_{n-m}。
- 使用矩阵乘法优化即可。

1406D（#）

- 不妨把 c **全部取个负**，变成 b,c 都不下降序列，a_i=b_i-c_i。现在答案为 max(b_n,-c_1)。
- 两个序列都要不下降，不妨 a,b,c **差分**一下。现在要求 a_1=b_1-c_1，A_i=B_i-C_i(i<=n-1)，B,C 非负。
- 答案是 max(b_1+B_1+B_2+…+B_{n-1},-c_1)。可以发现 B 中数字之和最小为 A 中非负整数之和，设其为 k。
- 现在变为 max(b_1+k,-c_1)，两数之和固定为 a_1+k，要求最大值最小，分析一下答案为(a_1+k)/2 上取整。操作只会修改两个位置，更新一下 k 即可。

837D

- 设 dp[i][j][x]表示考虑前 i 个数，当前选了 j 个数，5 的个数为 x 的情况下最多 2 的个数。
- 转移显然是 O(1)的，枚举一下当前数选不选，状态前两维分别是 n,k，第三维不超过 18n，过大 2 就不够了。

1392E（#）

- 若路径的和为 x，把 x 二进制下的 0/1 分成若干个连续段，如果段切换则走到下一行，如果不是则往右走。
- 这个显然可以构造出来：若是奇数行则全为 0，若是偶数行则是 2^(i+j-2)。
- 显然对于同个 x 不存在多种方案，求方案就模拟第一行的过程。

1450C2（#）

- 如果所有点都是关键点，显然有 3 种简单的染色。把**点分成(i+j)%3=0/1/2 三组**。
- 可行的染色方案：一组为 O，一组为另一种 X，剩下的一组不用改变。
- 求关键点的方案可以是它们求交，三种总操作次数为 k。根据鸽巢原理，必定有一个不超过 k/3。

1092D1

- 和 1332E 类似，由于给每个格子不停的+2，所以可以将初始的 a 都 mod 2，之后操作都**在 mod 2 意义下**。
- 你从前往后扫，如果**相邻的两个 a_i 相同，其实可以直接删除**，因为它们可以同时改变成 0/1。
- 最后剩不超过 1 个元素即可（其它数都可以变成它）。

1419E（#）

- 观察一下样例，n=pq（p,q 是两个不同质数）需要加数，其它情况好像都不要加。
- n=pq 比较简单，其它情况大概是 p_1p_m,p_1 倍数,p_2p_1,p_2 倍数,…,p_mp_{m-1},p_m 倍数。

546E

- 只能移动到相邻的，初始是 a，结束要求是 b，考虑网络流建图。
- S 到所有 i 连 a_i，i+n 到所有 T 连 b_i，然后 i 向相邻点+n 连 INF，代表可以走过去。

1025D

- 二叉搜索树的中序遍历是原序列排序后的值。
- 先把原序列排序，当前考虑[l,r]，要枚举选的根，现在是 O(n^3)。
- 可以发现，这个区间的父节点要么是 l-1 要么是 r+1。设 L[l][r],R[l][r]表示考虑区间[l,r]，父亲为 l-1/r+1 的答案。
- 转移依旧枚举[l,r]内的点当根，要求这个根和父节点的 gcd 不是 1。

1327D

- p^k 对应每个点向环中走 k 步到达的点连边，不过有**经典结论是分出的环长为 len/gcd(len,k)，个数为 gcd(len,k)，每个点往后跳 k 步形成的环和每个点往后跳 gcd(n,k)步形成的是相同的（第一张跳 1 次⇔第二张跳 k/gcd 次）**。
- 对于每个环，枚举 gcd 的值（一定是环长的约数），然后判断是否产生了至少一个同色环，可行的话就和 gcd 取 min（k 的最小值就是 gcd）。

1204D2

- **发现 10 是不能被其它等长的串所替换的**，它的最长不下降子序列长度为 1，而其它等长的串都为 2。如果 10 在序列中间，（设 1 的位置为 x）替换成其它的都会使得 len[x]>=len[x+1]+1， 而 10 的话 len[x]<=len[x+1]。
- 大胆猜测，就是把串中的 10 不断删除，直到不存在。
- 删除某个 10 后对所有 l,r 最长不下降子序列长度的影响:
  1. l,r 完全包含被删除的 10，那么原先无论最长不下降子序列是怎么选的，都一定会选择 10 中的恰好一个数，现在它的长度必定减少 1;
  2. l,r 和 被删除的 10 无交，那么最长不下降子序列长度不影响;
  3. l,r 和 被删除的 10 有交但不包含，即 l 或 r 在 10 之间，如果是 l 则原子序列一定会选择最左边的 0，如果是 r 则原子序列一定会选择最右边的 1，删去后长度必定减少 1。
- 删去后的问题，如果 s′ 和 t′ 满足条件，那么加回这个 10 后仍然满足条件。
- 所以可以一直删去 10，直到没有 10。考虑现在的串一定为 00...011...1，可以把所有数字都变成 0，所有 l,r 最长不下降子序列长度不变。
- 具体实现可以从左到右扫描，维护一个栈记录之前还没被删去的 1 的位置，如果当前为 0 且栈不为空则可以和前面的一起删去，否则加入栈。最后栈中的 1 即没有被删去的 1，都变为 0 即可。

1553E

- 有 m<=n/3 的条件，交换一次最多使得 2 个数归位，那么方案若可行**至少有 n/3 个数在原位**。
- 对于每个位置，只有一个 k 使其在原位。那么只有最多 3 个 k 是可能可以的，对于这些 k 暴力判断即可

1389E

- 由题意列出同余方程 dx+y≡dy+x(mod w)，移项得到 (d−1)(x−y)≡0(mod w)
- 将 w 去掉 gcd(d−1,w)得到 w′，于是 x≡y(mod w′)。
- 题目变为在[1,min(d,m)]里找两个模 w′同余的数。

- 把位置分成若干个同余类，组合数求解即可。

1338C（#）

- 打表。可以发现 a 在[2^{2i},2^{2i+1})中依次增加，而对应的 b,c 在[2^{2i+1},2^{2i+2})中。
- 进一步发现相邻四个 b 满足 0231 的关系，c 满足 0312 的关系。
- 再进一步发现对于 a 在一个块内的，b,c 先分成四小块，大小顺序满足 0231,0312 再递归下去分。
- 确定了 a 在哪一个块之后就可以暴力了。

1567E

- 线段树。左边和右边分别统计，然后再统计左边的后缀和右边的前缀形成的（左边的最右<=右边的最左）。
- 记录一下总和，长度，前缀长度，后缀长度，最左边的值，最右边的值。

1271E

- 不妨只考虑答案是奇数的情况。那么 path 中包含 x(x 是奇数)的数的个数即以 x 为前缀的数的个数。
- 设 n 的最高位为 l，令 m=n-2^l+1（2^l..n 之间的数字个数）。
- 枚举 x 的最高位为 i，现在要在 x 后面加后缀使得不超过 n。令 t=2^{l-i}。
- 首先，加的后缀长度不超过 l-i 一定是可以的，方案数为 2^0+2^1+…+2^{l-i-1}=2^{l-i}-1=t-1
- 如果 t-1+min(m,t)>=k 那么说明答案为 2^i 是可行的，接下来看看能不能更大。
- 一种情况是 t-1>=k，那么怎么取都一定>=k，答案是 min(n,2^{i+1}-1)。
- 还有一种是，前面压上界，那么要求 t-1+m%t>=k，如果可以则答案是 min(n, 2^i+m/tmp)。
- 否则不能压上界，答案是 min(n, 2^i+m/tmp-1)。

- 现在考虑答案是偶数的情况。那么 path 中包含 x(x 是偶数)的数的个数即以 x/2 为前缀的数的个数-1。
- 那么将 k+1，x/2，再按奇数的方法做一下，最后答案*2 即可。

1466F

- 考虑直接建图，对\sum a_i=2 中 1 的位置之间连一条边，对\sum a_i=1 的，让 m+1 和 1 的位置连边。
- 现在需要一张新的图，使得原来连通的两个点依旧能连通，不能成环。
- 那么相当于取个最小生成树，要求边的编号最小。那个所谓的|T|就是 2^|A|。

1305E（#）

- 考虑**先构造最大的情况**，固定了 k 的话，那么由于前面只有 k-1 个数，最多配成(k-1)/2 对。
- 达到这个上界可以构造 a_i=i，找到最前面的位置 x，使得按照 a_i=i 构造，到当前位置的值就>=m 了。
- 现在要增加 x 位置的值使得 m 减小，可以直接暴力。
- 至于后面的位置，要让它们不影响答案。设当前填过的最大数为 v，那么从后往前，每次填的数-(v+1)即可。

1254B2（#）

- 只需要考虑 k 是质数的情况。
- 其实 **a_i 和 a_i mod k 是完全等价的**（不会出现负数啥的情况），那么**前面往后移和后面往前移也是等价的**，计算一下答案即可。

1151E

- 有一种比较通用的做法：**点减边=连通块数**。
- 一个点被算进的次数是 l<=a_i,a_i<=r，l 有 a_i 种，r 有 n-a_i+1 种，总共 a_i*(n-a_i+1)种方案，求和。
- 一条边被算进的次数是 l<=a_{i-1}<=r,l<=a_i<=r，l 有 min(a_{i-1},a_i)种，r 有 n-max(a_{i-1},a_i)+1 种，总共 min(a_{i-1},a_i)*(n-max(a_{i-1},a_i)+1) 种方案，求和。
- 最后两者相减即可。

- 还有一种做法就是以每个连通块的最左端/最右端作为关键点统计。
- 考虑统计最左端，那么就是 i-1 不在[l,r]内，i 在[l,r]内，分类讨论一下。

1515E

- 这个题非常强大。
- 首先可以发现，你如果顺序存，需要知道中间连续段的长度，这个比较困难。
- 那么你考虑能不能不存，但是不存的话你不知道放的方案数有多少。
- 这题的关键在于，你不能直接决策在哪个绝对位置开了电脑，要决策相对位置（？）

- 不过现在有多少电脑开了还是得知道的，所以你考虑设 f[i][j]表示开了 i 台电脑，有 j 个开的连续段。
- 然后分类讨论
  1. 开中间，f[i][j]*(j+1)->f[i+1][j+1]
  2. 贴着开，f[i][j]*j*2->f[i+1][j]
  3. 隔一个开，f[i][j]*j*2->f[i+2][j]
  4. 两段间长度为 2，f[i][j]*(j-1)*2->f[i+2][j-1]
  5. 两段间长度为 3，开中间，f[i][j]*(j-1)->f[i+3][j-1]
- 答案为 f[n][1]。

839D

- 枚举 gcd=g，那么要求 gcd=g 的序列的长度和，再乘上 g 累加。
- 但是 gcd 恰好等于 g 的不好算，先求出 g|gcd 的长度和，再容斥。
- 从大到小求出恰好等于 g 的答案，如果是 g 的倍数的数有 x 个，那么长度和为 x*2^{x-1}，再减去是它的倍数的长度和，就得到了恰好的答案，*g 累加即可。

1166E（#）

- 看上去很难下手，先找点一定不合法的情况。
- 可以发现**两个限制没有交的话显然不合法**。
- 如果都有交，我给每个区间乘上一个不同的 p_i，那么区间的 lcm 是所有 p 之积，而补集一定没有 p_i，更小。

632D

- LCM 不超过 1e6，可以把 a 中>lcm 的删掉。
- 拿个桶存剩下的数的个数，枚举倍数，所有它的倍数的答案都加上这个数的个数。
- 最后输出答案最大的数（相同取最小的，否则可能出现 lcm 是它的约数）和是它约数的位置即可。

1451E2（#）

- 先假设序列为 0,a[1]^a[2],a[1]^a[3],...，实际的序列为每个位置异或上一个数 v。
- 只需要确定任何一个位置的值就可以求出原序列。
- 如果现在的序列有两个值相同，则问一下它们的 and 就可以知道它们真实的值。
- 如果没有相同，说明 0,1,...,n-1 都出现了一次，找到现在序列的 0 和 n-1 的位置，它们对应实际的值是 v,(n-1)^v，再找另一个位置 x，询问一下这两个位置和 x 的 and，or 起来就是 x 位置真实的值。

547A

- 你先暴力跳，把第一个跳满足，如果跳了 m 次还不满足就无解。
- 如果此时第二个也满足就直接结束。
- 否则你让第一个就再跳一个环回，跳回来，记录此时第二个乘的 x 和加的 y。
- 之后再跳第一个就不变了，依旧暴力跳，不超过 m 次。

1265E

- 考虑设置状态 dp[i]为从 1 到 i 的期望次数，有 dp[i]=pi*dp[i-1]+1+(1-pi)*(dp[i-1]+1+dp[i])=(dp[i-1]+1)/pi。

372C

- 设 dp_{i,j}表示当前在第 i 时刻，位置在 j 的最大收益。
- $dp_{i,j} \leftarrow dp_{i-1,k} \quad (k \in [j-(t_i-t_{i-1})\times d, j+(t_i-t_{i-1})\times d])$
- 发现同一层转移的区间长度是一样的，单调队列即可。

1097D

- 这题显然可以把每个质因子的幂次分开考虑。
- 设 f[i][j]表示 p^i 经过 j 次操作的期望，转移等概率变成 p^0,...,p^i，j-1。

- 可以直接记忆化搜索。

718C

- 矩阵是有分配律的，即 ab+ac=a(b+c)。
- 可以用线段树区间乘矩阵的幂次，维护区间和解决。

766E（#）

- 常规套路**枚举每个二进制位**。
- 记 dp[u][0/1]表示节点 u，路径异或值为 0/1 的路径数量（一段在 u 一段在子树内）。

1606E

- 设 f[i][j]表示存活 i 个人，每个人都扣了 j 的血量，已经死了的血量有多少种不同的方案。
- 那么 j 转移到的就是 min(i+j-1,x)，枚举死了多少人 k，就是血量在(j,min(i+j-1,x)]里的死了，组合数 C(i,k)选一下死的是哪 k 个，每个有 min(i+j-1,x)-j 种血量方案，幂次算一下即可。

1334E

- 考虑路径会长成什么样，直观的想法是先除后乘一定严格比先乘后除优，不妨反证。考虑从 x 到 x*p/q。
- 设 p,q 为两个不等质数，先乘后除：-d(x)+d(xp)+d(xp)-d(xp/q)，先除后乘：d(x)-d(x/q)-d(x/q)+d(xp/q)
- 要求-d(x)+d(xp)+d(xp)-d(xp/q)<=d(x)-d(x/q)-d(x/q)+d(xp/q)，整理得 2d(x)-2d(xp)+2d(xp/q)-2d(x/q)>0.
- 都除以 2，移项变成 d((x/q)p)-d(x/q)>=d(xp)-d(x)，可以知道，*p 之后增加量应该数字除去所有 p 之后，剩下的因子数，(x/q)|x，这个式子一定不成立。（为啥不考虑 p=q 呢，因为边权一定为正，没必要从 x 到 x）。
- 所以一定会先除后乘，否则可以交换顺序。那么可以变成 x 一直除，y 一直除，到达同一个点。
- 由于你只是除，那么路径上的中间点的权值都会删去，所以路径长度一定是 d(x)-d(目标点)。
- 那么答案是 d(x)+d(y)-2d(目标点)，x,y 都能到达的点只能是 g=gcd(x,y)的约数，可以发现 g 的约数中，g 本身的 d(g)是严格最大的，所以目标点也固定为 g。答案就是 x 到 g 的方案数*y 到 g 的方案数。
- 相当于要求一个数 x，每次除一个质因数，有多少种不同的方法，那么就用 tot!/(p1!p2!...pk!)算算多重组合。
- 由于题目说了给的都是 D 的约数，先求出 D 的所有质因子，询问的时候就可以 O(log V)解决。

280C

- 期望具有线性性 E(X+Y)=E(X)+E(y)，这个式子是不要求独立的。
- 你考虑每个数被删掉的概率，加起来就是期望。
- 第 u 个位置被删掉的概率，只和它到根节点路径有关系，和它没关系的可以直接不用考虑，删这条路径的时候第一个删它即可，那么概率为 1/dep[u]。
- 最后累加即可。

547C

- 先预处理每个数的质因子。现在要求 gcd 为 1。

- 2^7 暴力容斥有哪些质因子在 gcd 中，计算一下，带上容斥系数，先去掉原先贡献再更新，最后加上当前贡献。

1542D

- n 很小。考虑枚举每个+x，最后没被删除的方案数。
- 设 dp[i][j]表示前 i 个位置，有 j 个<(x,pos)（进行二元组比较，相同先删前面的）还存在的方案数。
- 转移如果是-，那么从 dp[i+1][j] += dp[i][j]（不加这个数），dp[i+1][j-1] += dp[i][j]（加这个数）。
- 如果是+，且加的数<(x,pos)，dp[i+1][j+1] += dp[i][j]（加这个数），dp[i+1][j] += dp[i][j]（不加这个数）；否则 dp[i+1][j] += dp[i][j]（加这个数），dp[i+1][j] += dp[i][j]（不加这个数）；
- 最后将 dp[n][>=0]加起来即可。

1154G

- 拿个桶记录每个数有没有出现过。
- lcm 可能很大，但 lcm(a,b)=ab/gcd(a,b)，考虑枚举 gcd 和它的倍数，求两个最小的出现过的更新答案即可。

1371E2

- 一个求和，一个整除，看上去不是很相关。先推推 f(x)的式子。
- 假设 1..i-1 都打过了，要判断第 i 个能不能打过，可以通过 x+i-1>=a_{P_i}来判断，移个项变成 x>= a_{P_i}-i+1。
- 那么不能有 a_{P_i}-i+1>x 的 i，贪心的来看从小到大取是最优的。
- 把 a 从小到大排序，x 不能小于 a_1，否则 f(x)=0；x 不能大于等于 a_n，否则 f(x)=n!，而 p<n 一定整除。
- 对于其它的 x，设 t_i 表示<=i 的数字个数，那么$f(x)=\sum_{i=x}^{x+n-1} t_i-(i-x)=\sum_{i=x}^{x+n-1} x-(i-t_i)$
- 预处理 i-t_i，最后判断是否有与 x 同余的即可。

1342E

- 一共只有 n 辆车，我们想让其中一对能互相攻击到，就需要将其中一辆车移到另一行或另一列，那么这辆车原来所在的行或列就会空出来，又要使每个空格子被攻击到，那么就要**保证每一列/每一行都至少要有一辆车**，所以计数的时候让每辆车对应一列或一行。
- 如果恰有 k 对车能相互攻击，当且仅当有 k 行没有车或者 k 列没有车。那么问题就转换为了有恰好 k 行或 k 列没有车的方案数。
- 这两个问题对称，所以只考虑行的情况，答案就是其两倍。（特判 k=0,k>=n）。
- 剩下的就比较套路了，计数将 n 个元素划分到 k 个非空集合的方案数，容易发现这就是第二类斯特林数。

1511E

- 横着的和竖着的是独立的，如果有一段长度为 x 的横/竖的极长连续段，设它的答案为 f[x]，那么对应的贡献为 f[x]*2^{tot-x}（tot 是总白色格子数量，即剩下可以任意选）。
- 考虑 f[i]的转移：
  1. i 位置是另一种颜色，从 f[i-1]转移过来
  2. i 位置是这种颜色，i-1 不是这种颜色，从 f[i-2]转移过来
  3. i 位置是这种颜色，i-1 也是这种颜色，从 f[i-2]+2^{i-2}转移过来（后面加的是之前的方案数）
- 横竖都算一边加起来即可。

785D

- 枚举最后一个左括号出现的位置 i。设 a_i 表示 1..i 中左括号数量，b_i 表示 i..n 中右括号数量。
- 那么，方案数量为$\sum_{j=0}^{+\infty} C(a_i-1,j)\times C(b_i,j+1)=\sum_{j=0}^{+\infty} C(a_i-1,j)\times C(b_i,b_i-1-j)=C(a_i+b_i-1,b_i-1)$。
- 只枚举左括号的位置即可。

1415E（#）

- 相当于**分成最多 k+1 个序列**，每个序列再按照顺序计算。
- 可以贪心考虑，显然分出的每个序列都是从大到小排序最好。
- 用堆维护每个序列当前的和，找到最大的序列加上当前的数。

1182E

- 不难发现，每个 f_i 可以表示成 c^{w_i}*f_{i-1}^{x_i}*f_{i-2}^{y_i}*f_{i-3}^{z_i}。
- 除了 w 是前面三项和+2i-6，其它都是直接前面三项和。矩乘即可（注意模数要-1）。

955C

- 注意到 p>=3 的话 a<=1e6，而且由于 a^p<=1e18，直接暴力枚举 a，暴力枚举 p 即可，求出所有不超过 1e18 的数后排序去重，询问时二分。
- p=2 的数可以直接用 sqrt(r)-sqrt(l-1)得到，但是可能有数即能被 p>=3 表出，也能被 p=2 表出，所以在 p>=3 表出的序列里，去掉 sqrt 是整数的数即可。

1526D（#）

- 根据经典套路，**存在一种最小代价最大的方案，相同的数字都排在一起**。
- 4! 枚举四种字母的顺序，树状数组求值即可。

768D

- 设 dp[i][j]表示 i 天恰好取了 j 种不同物品的概率。
- 那么有 j/n 的概率取了和之前相同的，有(n-j)/n 的概率取了和之前不同的，分别转移。
- 根据经典结论，概率为 p 的东西期望 1/p 次取到，求和一下，得到第一维是 n ln n 级别。

1536E

- 你考虑确定了哪些位置是 0（如果有 0）。显然的结论：**那么和 0 距离为 i 的位置填的至少为 i**。
- 那么和这些 0 相邻的且没填的位置只能填 1（填更大不满足条件 1），把 1 填上。根据结论不会有其它的 1。
- 和这些 1 相邻的且没填的位置只能填 2（填 1 的话相邻没有 0，不满足条件 2）。根据结论不会有其它的 2。
- 依次类推可以证明和 0 距离为 i 的位置填的只能为 i。此时也一定满足题目条件。

- 那么只需要算一下填 0 的方案数即可，即 2^cnt-[cnt==n*m]，cnt 即给定矩阵#的个数，注意特判初始没有 0，那么必须有 1 个 0（否则最小值一定不满足条件 2）。

1202C

- 行列独立，分别处理，以行为例：先求不操作得到的最小/最大值。
- 现在要让最大值-1（或最小值+1）,显然当当前值第一次等于最大值（最小值）的时候必须是由-1（+1）。
- 然后更新操作后的最小最大值，列同理。再计算一下。

985D（#）

- 两种可能。
- 一种是 x,x-1,...,1 和一个单独的数 t(1<=x<=h,1<=t<=x，也可能没有)。
- 一种是 h,h+1,...,x-1,x,x-1,...,1 和两个单独的数 p,q(x<=h,1<=p,q<=x，也可能只有一个或没有)。
- 二分或解方程即可。

525E

- 由于 19!>10^16，所以你不会给>=19 的阶乘。
- 暴力的复杂度是 O(3^n)，你要决策每个位置不选/选且不阶乘/选且阶乘。
- 这显然太慢了，由于要求的是和恰好为 s，考虑 meet in the middle。
- 先搜前半部分，加入哈希表，搜后半部分的时候查表即可。

1398E

- 设电的数量为 x。可以发现，理想的情况是你给最大的 x 个翻倍。
- 但是有可能不行，因为可能最大的 x 个都是电，这样一定有一个翻不了，可以发现一定不翻最小的。
- 而且，如果最大的 x 个中有火，你就从一个不用翻的电开始就可以了，此时一定没有最小的电。
- 所以无论什么情况，都不会翻最小的电。你要求的是答案尽量大。
- 如果 x 大有火则为 S+S1，否则为 S+S2-M（S 为所有和，S1 为前 x 大和，S2 为前 x+1 大和，M 为最小的电）。
- 可以用 set 维护前 x+1 大，相同的认为火更大。

487C（#）

- 设 b 为 a 的前缀积，若 a_i=n 则 b_i,...,b_n=0，所以 a_n=n；若 a_i=1 则 b_i=b_{i-1}，所以 a_1=1。
- 再分析一下发现对于 **n>4 且 n 不是质数的时候(n-1)!=0 mod n**，说明无解。
- 质数可以构造前缀积为 1,2,...,n，求逆元即可。

1537F（#）

- 这种就考虑生成树或二分图。按照经典套路，每次加或减的都是一个偶数，所以如果和的**奇偶性不对显然无解**。
- 可以发现，能把两个距离为奇数的位置一个+1 一个-1，能把两个距离为偶数的位置都+1/-1。
- 如果有奇环，因为原图连通，故任意两点都存在长度为奇数/偶数的路径，故一定有解。

- 如果没有奇环，那么是二分图，那么左部之和减右部之和不变，判断一下即可。

1270E（#）

- 按照经典套路，**考虑奇偶性分组**。按照 x 坐标奇偶性，y 坐标奇偶性拆成四个组。
- 同一个组的距离的平方都是 4 的倍数。接下来分类讨论。
- 如果 0,0 和 1,1 至少有一个非空，0,1 和 1,0 至少有一个非空，那么 0,0∪1,1 为一组，0,1∪1,0 为一组。
- 否则如果 0,0 非空且 1,1 非空，或 0,1 非空且 1,0 非空，那么分开即可。
- 如果只有一个组非空，把横纵坐标/2 继续递归即可，显然递归层数不会超过 log。

1461E

- 开始的时候 k 在[l,r]内。分类讨论。
- 如果 x>=y，一天过后水不会变多，第一天特殊考虑，剩下每一天一定会加水（尽量大，一定不会超过 r）。
- 如果 x<y，那么就是一直倒水直到再倒要<l 了，就加水。看上去这种复杂度很高，不过可以发现，再倒要<l 的状态只有 x 种，即 l,l+1,...,l+x-1，如果出现环（即状态重复出现）那么就一定能无限进行下去，否则每个状态最多遍历一次，复杂度 O(x)。

1158B（#）

- 考虑一种构造是 1,m 个 0,1,m 个 0,...,1,k(0<=k<=m)个 0。
- 串长 l(m+1)+1+k，最短唯一子串为(l-2)(m+1)+3+k。（形如 0,1,m 个 0,...,1,k 个 0,0/1）
- 发现总长比最短子串长 2m。m=(n-k)/2，那么总长就比最短子串长 n-k。
- 但是有个问题，如果原串的长度比 2(m+1)要小，此时最短唯一子串长度为 2。（形如 0,1）。
- 而 n<2(m+1)即 n<n-k+2，说明只有 k=1 才会出现这种情况，特判即可（形如 1,n-1 个 0）.

908D

- 设 dp[a][x]表示有 a 个 a，x 个 ab 的概率。
- 那么有 p 的概率转移到 dp[a+1][x]，1-p 的概率转移到 dp[a][x+a]。
- 但这不好，第一维可能无限大，且初值从 dp[0][0]开始可能有环。
- 初值问题好解决，改成从 dp[1][0]开始，那么概率依旧是 1。
- 第一维的问题，如果两维之和>=k，那么只要有一次加了个 b 就结束了。
- 不妨算一下，p^0(1-p)的概率为 k，p^1(1-p)的概率为 k+1，p^2(1-p)的概率为 k+2，以此类推。
- 由于总概率和为 1，不妨把 k 提出来，那么期望为$k+\sum_{i\ge 0}p^i(1-p)i=k+\sum_{i\ge 1}p^i=k+\frac{1}{1-p}-1$。
- 即你在两维之和为 k 的时候这么计算即可。

431D

- 考虑固定了 n 求个数，这个可以用数位 dp。
- 可以发现，每次区间是不断边长的，当 n->n+1 时，2(n+1)完全替代了 n+1（1 的个数一样），中间还增加了数，故值一定更大。
- 所以可以二分这个 n 用数位 dp 判断。

1477C（#）

- 如果构造的方案存在钝角∠ABC，那么根据**大边对大角**，AC 一定严格比 AB 和 BC 长。
- 所以如果 AB 比 AC 长就不存在这个问题，每次取离 A 最远的 B 即可。

1228E

- 刚开始想的是钦定多少行多少列不合法，后来发现没有必要，你只有一维是很好做的。
- 考虑**钦定 c 列不合法**，那么每一行除了 c 列外必须有一个为 1，这个直接减去都不为 1 的方案即可。
- 即枚举 c，组合数 C(n,c)选不合法的列，这些列有(k-1)^{cn}种方案。对于每一行，方案是(k^{n-c}-(k-1)^{n-c})，外面带一个(-1)^c 容斥系数即可。

601B

- 这个什么常数就是任意两点斜率最大值，而**任意两点斜率最大值就是相邻点斜率最大值**（证明考虑它类似于求差分数组的平均值，那么一定比最小值大比最大值小）。
- 设 d_i 为|h_i-h_{i+1}|，所以[l,r]的常数，就是[l,r-1]的 d 的最大值。
- 那么就是区间最大值之和，不妨钦定相等的情况下左边的更大，算一下每个点为最大值的区间。
- 枚举最大值位置，知道区间的做右端点就可以把两边长度（i-l+1,r-i+1）乘起来求和，就是答案。

571A

- 合法需要满足任意两边之和大于第三边，即三个条件同时满足。
- 而不合法即存在一边不小于另两边之和，即三个条件满足任意一个，且三个条件最多有一个满足。
- **补集转化**，求不合法的。总方案数显然可以隔板法即 C(l+3-1,3-1)。
- 剩下就是枚举哪个不小于另外两边之和，再枚举它加了多少，另外两个依旧用隔板法算。

1166D

- 你考虑前缀和，x_i=S_{i-1}+r_i，那么 S_i=2S_{i-1}+r_i。
- 那么你要 S_{n-1}+1<=b<=S_{n-1}+m。
- 你先让 S 尽量小，然后再反过来加大，先给 S_{n-1}加大同时满足条件，把剩下的给 S_{n-2}，再不停的往前推。
- 好像和题解写的不太一样（？）

1559E

- 容斥 gcd 的条件，求每个数是 x 的倍数的方案数。
- 剩下的类似背包即可（区间转移），复杂度是调和级数，O(nmlnm)。

616E

- $n \bmod i=n-[n/i]*i$，对$[n/i]$整除分块，$*i$ 的话算 $l+(l+1)+\dots+r$ 的和即可。
- 时间复杂度 $O(\sqrt{n})$

1562D2

- 设 $sum_{l,r}=\sum_{i=l}^r (-1)^{i-1}a_i$。
- 首先特判 l,r 本身和即为 0，否则如果 $r-l+1$ 是偶数至少需要两次操作，是奇数至少需要一次操作。
- 考虑一次操作删除了位置 p，那么 $sum_{l,p-1}-sum_{p+1,r}=0$，即$(pre_{p-1}-pre_{l-1})-(pre_r-pre_p)=0$。
- 移项得 $pre_{p-1}+pre_p=pre_{l-1}+pre_r$，只考虑 $r-l+1$ 是奇数，那么奇偶性一定对。
- p 变成 $p+1$ 的时候，左边的值可能-2/不变/+2，讨论一下 $p=l$ 和 $p=r$，发现大概是连续的，中间的一定能取到。
- 询问的时候在 vector 二分这个满足条件的 p 即可，长度是偶数先删掉 l 即可。

1540B

- 先固定一个根表示第一个选的，最后求个平均值。
- 把题目转化为 a,b，$a>b$ 且 a 在 b 前面的概率，求和即可。
- 那么这个概率显然和 a,b 路径以外的点无关，只考虑 a,b 各自到 LCA 的距离，分别记为 x,y。
- 求出 $f(x,y)$，则 a 在 b 之前的概率为 $f(x,y)$，$f(x,y)=1/2f(x-1,y)+1/2f(x,y-1)$。
- 枚举根复杂度 $O(n)$，枚举 a,b 复杂度 $O(n^2)$，求 LCA 复杂度 $O(1)/O(\log n)$（$O(1)$可以通过预处理）。
- 总时间复杂度 $O(n^3)$或 $O(n^3\log n)$。

980D

- 任意两数是平方数这个条件不太好处理。不过可以发现 $ab=n^2,bc=m^2 \Rightarrow ab^2c=(nm)^2$。
- 如果 $b=0$ 就比较麻烦，不过发现原题 $a[i]=0$ 的位置不影响答案（不过要注意 a 全为 0 的情况）。
- 现在 $a,b,c\neq 0$，那么 $ac=(nm/b)^2$，而 ac 是整数，那么 nm/b 也是整数，则 ac 是完全平方数。
- 那么可以使用并查集将两个乘起来是完全平方数的合并，现在就是对每个区间看有多少个并查集。
- 直接枚举 l，向右移动 r，同时看是不是属于新的并查集即可（0 就不用管）。

402E

- 首先这个 $a_{i,j}>0$ 的都可以看成 1，因为 1 和其它正数没有区别。
- 那么 1 次就可以看作图的邻接矩阵，2 次可以看作图上 u 走 2 步能到达的点，...
- 题目变成任意两个点都有长度为 k 的路径互相到达。
- 先考虑处理任意两点都能互相到达，可以发现 u 能到 v，v 能到 u 即 u,v 在同一个强连通分量。
- 即所有点都是强连通的，现在要考虑长度为 k 这个条件。由于强连通所以任意两点都有长度$<n$ 的路径。
- 题目给了 $a_{i,i}>0$，那么每个点都有自环，当 $k=n$ 时就一定满足题意。
- 所以只需要判断是否所有点都强连通，tarjan 缩点即可。

549H

- 答案具有可二分性，即二分 mid，判断四个数绝对值差<=mid 是否可行。
- 大概估计一下，这个 ad 和 bc 的取值是连续的，求出 ad,bc 在固定 mid 时可能的最小最大值。
- 那么一定在边界取到，即每个数都是-mid/+mid 之一，算一下。判断 ad 的区间和 bc 的区间是否有交。

1592D

- 一条路径的权值一定不如上面每条边权值的最大值来得大。所以只需要考虑每条边的最大值。
- 它把树给你了，你每次要询问一部分的点，得到它们之间边的权值最大值。
- 答案是 log n 级别的，考虑找到一个顺序，然后二分。考虑树的欧拉序。
- 你每次二分 mid，看[l,mid]的值是不是最大值，如果是就到[l,mid]，否则到[mid,r]（注意 mid 也要在内，因为没考虑 mid,mid+1 对应原图节点的边）。

776E

- f(n)=\sum_{i=1}^n [gcd(i,n)=1]=phi(n)（倒过来用一次更相减损法）。
- g(n)=\sum_{d|n}f(d)=\sum_{d|n}phi(d)=n（phi×1=id）。
- F_1(n)=phi(n)，F_2(n)=phi(n)，F_3(n)=phi(phi(n))，F_4(n)=phi(phi(n)),F_5(n)=phi(phi(phi(n)))以此类推。
- 即 F_k(n)即 n 做[(k+1)/2]次 phi 得到的值。
- 根据经典结论，phi 做 O(log n)次就变成 0 了（奇数变偶数偶数/2），所以可以直接暴力。

603C

- 显然不同石子堆是独立的，将 sg 值异或即可。
- 如果 x 是奇数，sg(x)=mex(sg(x-1))；如果 x 是偶数，sg(x)=mex(sg(x-1),sg’(x/2))。
- sg’(x)=sg(x)^sg(x)^...^sg(x)（k 个），即 k 是奇数为 sg(x)，k 是偶数为 0.
- k 是奇数则 x 奇数 mex(sg(x-1))，x 偶数 mex(sg(x-1),sg’(x/2))。
- k 是偶数则 x 奇数 mex(sg(x-1))，x 偶数 mex(sg(x-1),0)。
- 推出最前面的几个值，发现 x 是奇数值就是 0，x 是偶数递归即可。

1543D2（#）

- 每次操作可以看作是每位模运算下的减法 z=(y-x+k)%k，它**和模运算的减法一样**满足很多限制。
- 考虑枚举答案 i，判断初始的 x 是否等于 i。比如开始 i=1，x⊖1，然后判断是不是 0。
- 如果不是，现在要判断是不是 2，可以通过(2⊖1)⊖(x⊖1)=2⊖x，3 可以通过(2⊖3)⊖(2⊖x)=x⊖3 等等。

520E

- 首先一共有 n−1 个位置可以放加号，讨论 a_i 之后距离 a_i 最近的加号位置。
    1. 在 a_i 后面，那么还有 n-2 个位置可以放 m-1 个，是 a_i*C(n-2,m-1)。
    2. 在 a_{i+1}后面，那么还有 n-3 个位置可以放 m-1 个，是 a_i*10*C(n-3,m-1)。
    3. 在 a_{i+2}后面，那么还有 n-4 个位置可以放 m-1 个，是 a_i*100*C(n-4,m-1)。
    4. ...
    5. 特殊的，在 a_n 后面，那么还有 i-1 个位置可以放 m 个，是 a_i*10^{n-i}*C(i-1,m)。
- 直接枚举 i 和 10 的幂次 j 复杂度是平方的。
- 发现 j 相同的乘上的组合数和幂次都是相同的。直接算即可。

590B

- 只和起点终点的位置两维差有关。
- 两个运动是独立的，那么飞艇的运动可以拆成驱动运动和风力运动。
- 二分答案，看风力运动会运动到哪里，判断解是否可行即可。

1598E

- 刚开始处理出答案，询问的时候往左上走/往右下走扩展。
- 这题打了比赛，现场还想复杂了，想什么第 i 行往左移 i 位啥的，写的时候才发现没有必要。

1413E

- **发现如果 a>bc，那么他无论怎么恢复都不如减的多**，答案是无穷大。
- 否则 a<=bc，不妨假设 0 时刻你就攻击，到 a/b 上取整时刻，减的就比加的多了，不如第一个不做。
- 所以只需要考虑到[(a-1)/b]时刻，那么你一定是能攻击就攻击，在[[(a-1)/b]/d]次操作后达到最大值。
- 计算一下答案即可。

145C

- **幸运数个数很少**，总个数不超过 2^1+2^2+2^3+...+2^8+2^9 个。
- 而每种只能取最多一个，那么设 dp[i][j]表示前 i 个幸运数选了 j 个的方案数（同时要乘上 a 中位置数量）。
- 现在知道了幸运的选多少个，不幸运的组合数算一下即可。

486E

- 判断一个数能不能在 LIS 上：求出以每个数为开头/结尾的 LIS 长度，看看加起来-1 是否等于最长的 LIS 长度。如果等于，那么说明它在 LIS 上，且位置固定（它一定是 LIS 上第 x 个位置）。
- 判断一个数是不是必须在 LIS 上：看 LIS 第 x 个位置有没有多种选法，如果有就不必须在，否则必须在。

1073E

- 考虑数位 dp，solve(pos,sta,lim,lead)求出当前在第 pos 位，当前出现过的数的状态为 sta，是否压上界，是否有前导 0 的方案数/总和。
- 转移就直接枚举当前位填什么即可。

1553F

- 题目保证了数字两两不同，且值域很小。
- 当前枚举到 i，现在要求 a[j] mod a[i]与 a[i] mod a[j]的和(j<i)。
- a[j] mod a[i]：考虑枚举商 y（a[i]y<=n），那么[xy,(x+1)y)之内的 a[j] mod a[i]的值即 a[j]-a[i]y。
- a[i] mod a[j]：在做 j 的时候，枚举商 y，那么[xy,(x+1)y)之内的 a[i] mod a[j]的值即 a[i]-a[j]y，在树状数组上提前更新即可。

- 树状数组区间个数和区间和即可，总复杂度是 $O(n \ln n \log n)$。

1500B

- 反过来考虑那些 a,b 相同的，可以发现只有在 i%lcm(n,m)为某个值的时候才有可能发生。
- 先暴力，a_x=b_y，求出 q=lcm(n,m)的值，然后取模。
- 剩下来最多做不超过 lcm(n,m)，还是在 a_x=b_y 的 x,y 不会有贡献，算出它们对应的 i，排个序，判断一下即可。

1593F

- 设 dp[x][y][i][j]表示当前红色位置%a=x，蓝色位置%b=y，i 个红色，j 个蓝色是否可行。
- 转移枚举 a[i+j+1]当前涂什么颜色，分别转移即可。

551D

- 先特判 k>=(1<<l)的情况。全都是位运算，考虑每位分开考虑。
- 现在相当于 n 个数填 0/1，使得(a_1&a_2)|(a_2&a_3)|……|(a_n-1&a_n)=0/1 的方案数。
- 为 0 和为 1 的方案总数为(1<<n)，为 0 即不存在相邻两个都是 1。
- 考虑为 0 的，那么 f[n]=f[n-1]（当前为 0）+f[n-2]（当前为 1，前一个必须为 0）。
- 斐波那契数列，矩阵快速幂即可。设 f[n]=a，为 0 的位答案就乘 a，位 1 的位答案就乘((1<<n)-a)。

1027E

- 不妨钦定 a[1][1]=0，最后答案*2。
- 假设知道了 a[*][1]和 a[1][*]，那么就知道第 i 行和第 1 行是否相同，第 j 列和第 1 列是否相同。
- 具体地，a[i][1]如果是 0，那么说明和第 1 行相同；a[1][j]如果是 0，说明第 j 列和第 1 列相同。
- 更近一步地，a[i][j]=a[i][1]^a[1][j]。如果 a[x1..x2][y1..y2]同色，根据式子可以知道 a[x1..x2][1]同色，a[1][y1..y2]同色。
- 所以最大同色块即 a[*][1]的最长连续段长度和 a[1][*]的最长连续段长度之积。
- 两维分开计算出最长连续段长度=x 的方案数（显然是对称的），将乘积不超过 k 的乘起来相加即可。

919E

- n≡n+p(mod p)，a^{p}≡a^{n+p-1}(mod p)。
- 显然有循环节 p(p-1)，但这个循环节太大了。
- 由于指数上是 mod (p-1)，不妨考虑设 n=k(p-1)+r　(0<=r<p-1)。
- 那么 na^n=(kp-k+r)a^{k(p-1)+r}≡(r-k)a^r(mod p)。
- 枚举 r，(r-k)a^r≡b(mod p)，即 k≡r-b*a^{-r}(mod p)
- 算一下合法 k 的个数即可。

1179C

- 从大到小枚举最后剩下来的值 x 看看可不可行，可行当且仅当 **a_i>=x 的个数严格大于 b_i>=x 的个数**。
- 现在你要带修改，且要快速询问，保存 a_i>=x 的个数-b_i>=x 的个数，求最靠右的>0 的位置。
- 维护线段树，每次看右儿子的最大值，如果<=0 就到左儿子，否则到右儿子。

678E

- 设 f[s][i]表示只剩下 s 中的人，之前获胜留下的是 i，0 号胜利的概率。初始一定是 f[1][0]=1。
- 考虑 f[s][i]的转移，枚举下一个和它决斗的是 j，一种是它打赢了，从 f[s^(1<<j)][i]*a[i][j]转移；一种是它打输了，从 f[s^(1<<i)][j]*a[j][i]转移，看哪个 j 使得加起来的值最大即可。
- 答案是 f[(1<<n)-1][*]的最大值，你可以任意决定哪个是第一个。

1148E（#）

- 首先**总和不变**。可以把操作看成对于 s_j-s_i>=1（=1 相当于交换顺序，无关紧要）的两个数，s_j--，s_i++。
- 对 s_i++，s_j--，只要对值相同的取最后/最前的一个，相对顺序就不会改变，说明存在操作使**相对顺序不变**。
- 将 s,t 排序，设 a_i=t_i-s_i。现在要把前面的某个-1，后面的某个+1。可以发现有解相当于要求前缀和>0。
- 具体的构造方法类似于括号序，看栈底和当前数的大小关系即可。

992D

- 因为 maxn*maxv*maxk=2e18，说明总和不能超过 2e18，所以乘积中非 1 的数最多只有 log(2e18) 个。
- 记录 nxt[i]表示 i 后一个不为 1 的位置，最多只需要跳 61 次，故总复杂度是一个 log 的。

1183F

- 只选一个数显然**选最大**的。
- 选两个数的话讨论一下，如果不**选最大的**，如果剩下的有它的约数，不如选那个数；否则没有，最多 a/2+a/3<a。
- 选三个数依旧讨论，要么有约数，要么最多 a/2+a/3+a/5，特判一下，否则就一定**选最大**的数。
- log 的。

1267K

- 求出对应的集合，打乱，看**哪些打乱后的顺序可以恢复**。
  1. 最后一个数不能是 0，由于是余数，是 0 的话说明还没有除完或者上一次就是 0.
  2. 除数是从 2 开始的，所以第 i 个数不能>i。
- 满足这两个条件就都是合法的，可以还原出来。
- 第一个限制可以直接容斥，第二个限制从大到小枚举，计算一下能放的位置数量即可。

1550D（#）

- 把式子移项变成 a_i-i=-(a_j-j)，那么这些数的**绝对值一定相同**。
- 设绝对值为 x，那么有 n/2 个为+x，n/2 个为-x。（一个取上整一个取下整）
- l-i<=a_i-i<=r-i，大概考虑一下每个 i 能取+x 的最后时间，能去-x 的最后时间，再计算一下即可。

1252K

- 直接模仿 NOID2T2 先搞成矩阵形式，A=[1 1\0 1],B=[1 0\1 1]。
- 一种是和那题类似的，你用线段树记录原来的区间乘积和取反的区间乘积，修改可以直接打 Tag。
- 后来发现这题好像更简单，好像取反就是上下行交换然后左右列交换。

1525E

- 反过来计算每一个点的贡献。一个点是否被点亮和点亮城市的排列有关。
- 求出不能到达的方案，再用总方案减去。
- 考虑依次计算排列的第 i 个位置，那么它必须满足距离>n-i+1，计算一下。

712D

- 最后要求 a>b，即 a-b>0。
- 不妨设 c=a-b，每次操作都是+[-k,k],-[-k,k]，发现这个和做 2t 轮没有区别。
- 所以问的就是，初始是 a-b，做 2t 次操作>0 的方案数。
- 直接背包，第一维 O(t)，第二维 O(tk)，总复杂度 O(t^2k)。

1217E

- 有个简单的思路，平衡的是不是**每一位最多只有一个数非 0**。
- 思考一下，发现如果有多个数某位都非 0，且还要满足条件，那么一定需要进位。
- 对于前面的位，如果后面有进位，要满足条件自己也要进位。
- 到最后会进到一个没有数出现过的位，就 gg 了。
- 那么不平衡的一定有两个数在某一位都非 0，要求最小就是这一位非 0 的最小的两个数之和。
- 线段树记录一下区间每一位非 0 的数中，最小和次小的值即可。

995C

- 原来每个不超过 1e6，最后要求和不超过 1.5e6。
- 向量相加显然和夹角有关系，考虑哪些夹角可以。
- c^2=a^2+b^2-2abcosC，不妨设 a<=b，那么 c^2<=a^2(1-2cosC)+b^2，当 cosC<=1/2 时 c^2<=b^2。
- （这个是把两个向量首尾相接，形成三角形两条边）。
- 拿出三个向量，正反有 6 个向量，一定有两个夹角<=pi/3，满足条件。
- 到最后只剩下两个向量时，夹角之间<=pi/2，这样长度可能乘上 sqrt(2)，也不超过 1.5e6。

1158C（#）

- 先考虑没有-1 的情况，存在 i,j，**i<j<nxt_i<nxt_j 一定无解**。**否则可以构造出解**。
- 对于后缀最大值（nxt_i=n+1），它们之间的小区间又是独立的（nxt_j<=i），递归下去做。
- 具体实现可以类似于建树，i 的父亲是 nxt_i，层数低的比层数高的值大，同层编号小的比编号大的值大。

- nxt_i=-1 的情况，直接取 nxt_i=i+1 即可，这样不会使得原来有解的变成无解。

1139D

- E(x)=\sum_{i>=0} P(X>i)，变成求长度为 i 的序列 gcd 不为 1 的方案数。
- P(x>i)是好求的，枚举所有数都是 d 的倍数，然后莫反。
- 带回原式子，把 d 提到外面，里面是个无穷等比数列求和，就做完了。

1510D

- 是否合法可以把初始的所有数字都 mod 10 来判断。
- f[n][0..9]的状态，记录一下到第几位，mod 10 的值是多少，现在最大的乘积。
- 不过乘起来就太大了，一种方法是存 log 后的值。

- 除了 d=0，否则不存在 10|d，所以 2|d 则 5∤d，5|d 则 2∤d。
- 所以有四种情况：d=0，2∤d 且 5∤d，2|d 且 5∤d，5|d 且 2∤d。
- 而 2∤d 的，不能选 2|a[i]的数；5∤d 的，不能选 5|a[i]的数。
- 第一种要么选 0，要么选末位为 2 和 5 的至少一次。
- 第二种只有 1,3,7,9 四种数能选。
- 第三种必须有 1 个 2，枚举哪个提供了这个 2，剩下的直接在 mod 5 意义下做即可。
- 第四种必须有 1 个 5，枚举哪个提供了这个 5，剩下的直接在 mod 2 意义下做即可。
- 现在就有逆元了。具体做法是先求出所有数的乘积，反过来记录 f[n][0..9]为最小的乘积（乘起来的数不会超过 4 个）。

538E

- 考虑权值固定的做法，设 dp[u]表示现在在点 u 的结果，一个点是谁操作之和深度有关，不需要额外记录。
- 现在权值不固定，但是可以发现，dp[u]的结果只与叶子节点相对大小有关。

- 要求答案最大：设 dp[u]表示现在在点 u，得到的结果是第几大的权值。
- 如果是 A 操作，那么一定选 dp[v]最小的转移（可以给那个子树填最大的若干个）；如果是 B 操作，想让所有子树第 dp[v]大的最小值尽量大，那么要把最大的\sum dp[v]个填到这些子树，从\sum dp[v]转移。

- 要求答案最小：设 dp[u]表示现在在点 u，得到的结果是第几小的权值。
- 如果是 A 操作，那么一定选 dp[v]最小的转移（可以给那个子树填最小的若干个）；如果是 B 操作，想让所有子树第 dp[v]小的最大值尽量小，那么要把最小的\sum dp[v]个填到这些子树，从\sum dp[v]转移。

1468H（#）

- 首先每次减小 k-1 个，(n-m) mod (k-1)不为 0 显然无解。
- 设 d=(k-1)/2，考虑最后一步，中心是 b_x，删去了中心左边的 d 个数，右边的 d 个数。
- 如果不存在 x 使得 b_x 左/右边有 d 个非 b 中的数，那么显然无解。
- 否则你在前面删的时候，k 个数都取要删的数，**一定能调整成最后 b_x 左右各 d 个数**。

1603C

- 如果你知道一个数最后分成了 x 个，那么一定是尽量平均的分配（设 y=val mod x，则有 x-y 个 u 和 y 个 u+1，其中 u=[val/x]，无论怎么调整都不能使最大值减小或最小值增大）。
- 考虑怎么求整个序列的值，这就显而易见了，你从右往左，每次求最少操作几次，使得值不超过右边的数的值即可。
- 但是，你一个数拆分是整除的形式，即这个 u 只有 sqrt V 种，当前数尽量大，转移一下即可。

1411E

- **这种题目（什么能不能达到，能不能凑出来）考虑寻找不变量（即无论输入什么都不会变的）。寻找不变量之后考虑所有能到达的状态。**
- 对于这题，就考虑字符是什么和字符前面的符号无关，考虑哪些符号是可达的。
- 首先最后一个一定是+号，同样的，倒数第二个一定是-号。
- 考虑是否满足这个条件就可以了，找到一对+-，那么就可以从减号的位置递归。
- 这有一种情况可能找不到，就是------…-+，只有最后一个位置是加号，那么取 m=|S|-1 即可。
- 去除最后两个数的限制，之前的可以直接贪心，从大到小排序，一个一个加入。否则你后面取的一定有一部分是抵消它的（这个是 2 的幂次，一次只能在一位上加），可以全部取反。

538F

- 考虑在 k 叉堆中，u 的父亲是什么。当 k=1 时是(u-1)/1，k=2 时是 u/2，k=3 时是(u+1)/3，k=4 时是(u+2)/4，以此类推，可得对于任意的 k，u 的父亲是(u+k-2)/k。
- 一种想法是考虑整除分块，这个是 sqrt 的（考虑父亲）。
- 另一种想法是考虑从小到大加入，用树状数组维护每个位置有没有被加入，然后枚举它的倍数，看对应的区间里面的数字个数，即不满足条件的儿子数，这样是 log^2 的（考虑儿子）。

1032E

- 如果只有一种砝码，显然直接就知道所有的。
- 如果有两种不同的砝码，那么一定是可以一次就问出所有的（问质量大的和它的个数）。
- 固定了 k,w 之后，只有一种拆分方法，且所有砝码相同的才可以。看每一个 k,w 固定后有多少种拆分方法即可。

901B（#）

- 没系数和值域限制的话显然直接斐波那契数列就行了。
- 构造 p(0)=1，p(1)=x，p(n)=x*p(n-1)±p(n-2)。
- **但是无论是+/-，它们在 mod 2 意义下是一样的，且只要保证最高位是 1，就不会影响递归次数**。
- 直接 p_i=x*p_{i-1}+p_{i-2} mod 2 即可。

920G

- 首先可以转化成二分求 1..x 和 p 互质的数的个数。

- 由于 2*3*5*7*11*13*17*19>1000000，所以 p 最多只有 7 个不同的质因数。
- 2^7 枚举一下哪些质因数出现了，算一下个数，$(-1)^{二进制 1 的个数}$ 容斥一下即可。

346C

- 首先相同的 x_i 显然可以删去。2 操作相当于走到左边一个最近的 x_i 的倍数。
- 如果 2 操作某个 x_i 已经<b，那么之后操作这个 x_i 一定也<b，删去即可。
- 否则就需要决策怎么走是最优的，发现一定是走 2 操作后最小的 x_i，走到 a-a mod x_i，否则假设走另一个 x_j 更优，那么考虑某一步一定会从>=a-a mod x_i 走到<a-a mod x_i，那么这一步在 a-a mod x_i 开始也一定可以走，步数也不会更多。所以一定走 2 操作最小的。
- 算一下时间复杂度，你每两步操作就会使答案至少减少当前 x 的最大值，而 x 的最大值>=数字的个数，而你每次操作的复杂度也是数字的个数，所以复杂度和 a-b 是线性关系。

979D

- 把条件转化为 v<=s-x,k|v,k|x，要求 x xor v 最大。（可以直接判断 k|x 是否满足，不满足则无解）
- 由于 v<=10^5，且同一个数多次加入没有必要，所以可以考虑在加入的时候，给所有 k|v 的 trie 插入 v。
- 询问时只需要对 k 的 trie 查找，trie 每个节点记录子树满足条件的最小的 v 是多少（要满足 v<=s-x），看往哪个子树走。

1096F

- 分别计算-1 之间的贡献，-1 和数字的贡献，数字和数字的贡献。
- -1 之间的显然是 1/2，数字之间的可以使用树状数组计算。
- -1 和数字的，考虑枚举数字，看它左右-1 的个数，左边的形成逆序对需要大于它，右边的形成逆序对需要小于它，分别计算一下即可。

550E（#）

- 特判 n=1。如果**最后一个数是 1**，那么无论怎么操作，最后一个数的值一定是 1，**无解**。
- 如果倒数第二个是 1，那么前面怎么合并都是 1，最后和 0 合并即可。
- 000 可以变成 10，即只要最后 0 的个数=1 或>=3 都一定有解。
- 考虑最后 0 的个数为 2，即 xxx...1..100。如果之前还有 0，那么可以把 1..10 合并成 0，那么最后就有 3 个 0，有解；否则只能是 1..100，不能操作 00（最后一个数会变成 1），操作 11 或 10 都是减少 1 个 1，最后只能变成 00，00 显然没有合法方案。

1098B（#）

- 一种是隔行相同，隔列字符集合相同；一种是隔列相同，隔行字符集合相同。（或两者都有）
- 枚举一下是第一种还是第二种（2），枚举隔行/列相同的集合和顺序（4*3），枚举另一种的顺序（2），枚举另一种隔列/行变不变（2），然后算一下即可。

584E（#）

- 将 p 变成 1,2,...,n，**变成将 s 交换成升序的最小代价**。
- **答案的下界是|i-p_i|之和/2**，达到下界必须每次交换都没有浪费。
- 考虑从大到小确定数的位置，枚举 i 表示>i 的数都归位了，现在要将 i 归位。
- 找到 i 右边第一个 j 使得 p_j<=pos_i 的位置，交换。这样显然不存在浪费，最后也一定会归位。(没归位前右边一定有<=pos_i 的数）

266C（#）

- 递归，首先保证最后一列无 1，那么再交换有 1 的行和最后一行。
- 可以去掉最后一行，最后一列，再重复上面的步骤。

815B（#）

- 找规律，n%4=2 时，第 i 项对答案的贡献是 C((n-2)/2,(i-1)/2))。
- 暴力，直到 n%4=2 即可。

1117E（#）

- 由于 26^3=17576>15000。
- 先询问一次 aaa...aaa(26*26)bbb...bbb(26*26)ccc...ccc(26*26)...，知道在哪个大块。
- 再询问一次 aaa...aaa(26)bbb...bbb(26)ccc...ccc(26)...，知道在哪个小块。
- 最后问一次 abc...xyzabc...xyz...，知道在哪个位置。

837E

- f(a,b)=1+f(a,b-gcd(a,b))，设 a=xg,b=yg，那么 f(xg,yg)=1+f(xg,(y-gcd(x,y))g)，之后的每一步都乘了一个正整数 g，不影响答案。
- 所以 f(a,b)=f(a/gcd(a,b),b/gcd(a,b))，现在只需要考虑 a,b 互质的情况。那开始 gcd 是 1，可能减若干次就不是 1 了。所以要找到最大的 c 使得 c<b 且 gcd(a,c)>1，这样就能至少 a,b 都除以 2 了。
- 这个其实很简单，就找 a 的质因子最大的<b 的倍数即可。
- 求质因子是 sqrt 的，递归层数 log，总复杂度是 sqrt*log。

452C

- 首先特判 n=1。不妨设第一次抽到 1 号牌。
- 一种情况是前后抽的两张牌是同一张，这就相当于在 n 张牌中抽到那一张牌，概率为 1/n。
- 另一种情况，不是同一张牌（(n-1)/n），要在(nm-1)张抽(m-1)张 1 号牌.
- 两种情况相加即可。

1276C（#）

- 不妨假设 R<=C。那么选进来的数出现次数都不超过 R。
- 固定了 R 之后，每种数就尽量选，看最多能选多少，再确定 C。
- 构造的方法是不停地往右下走，如果走过了就再往右。

439E

- 这个直接考虑枚举 gcd 的约数 d（即每个数都是 d 的倍数）。
- 将所有数字/d，变成求和为 n/d 的序列数量，C(n/d+f-1,f-1)算一下，带上 mu(d)的容斥系数即可。

1548D1

- 由 pick 定理，S=A+B/2-1（A：内部格点，B：边上格点，S：面积）。
- 所有坐标都为偶数，可以把所有坐标/2，得到一个面积/4 的图形，而根据 pick 定理，多边形的面积一定能表示成 x/2（x 为整数的形式），所以原图形能被表示成 2x，即面积是偶数。
- 题目要求 A 是奇数，那么 B/2 一定是偶数，即 B 一定是 4 的倍数。
- 对于一条直线(x1,y1),(x2,y2)，它的点数是 gcd(|x2-x1|,|y2-y1|)。
- 发现可以 mod 4 之后来做，之后就可以暴力枚举三角形。

463E

- 改变的次数很少，且 2*3*5*7*11*13*17*19>2000000，考虑修改后暴力求出整个树每个点的答案。
- 相当于求祖先中最深的不互质的数，由于不同质因子只有 7 个左右，可以直接维护一个 vector 记录一下。

1042E

- 按照权值排序，直接暴力 dp，设 dp[i]表示从 i 开始的期望得分，枚举 j，然后转移。
- 把平方的贡献拆开，记录一下前缀 x^2 和，y^2 和，x 和，y 和，f 和即可。

1039C

- 对于所有边是否安全都没有关系的 x，贡献一定是 2^n（每个点都可以取或不取）。
- 否则 x 一定有 l,r 满足 c_l^x=c_r，那么这样的 x 只有 O(m)种，且这两个点只能都选或都不选，对同样的 x 一起合并，最后求的是 2^{剩下点个数}。

1167F

- 对于所有 i，包含 i 的区间至少有 1 的贡献。对于 i,j，a_i>a_j，包含 i,j 的区间会使得 i 多 1 次贡献。
- 包含 i 的区间个数为 i(n-i+1)；包含 i,j（a_i>a_j）的，不妨设 j<i，那么求的是 j(n-i+1)，即之前小于 a_i 的数字下标之和乘上(n-i+1)。
- 数字下标之和可以从左往右权值树状数组，对于 j>i 也类似。

1204E

- 这个看上去就很折线法，不妨考虑钦定最后的最大前缀和要>=v。
- 每次往右上或右下走一步，最后要走到 n-m，要求必须经过 y=v，求方案数。
- 如果本身 n-m>=v 就直接 OK，否则将经过 y=v 的路径对称一下，变成走到 2n-2m-v。
- 注意这样求得是前缀和>=v 的方案数，最后求和才是答案。

1513E（#）

- **如果比平均值大，就一定是源，否则比平均数小只能是汇**。和平均值相同的随意（无法操作）。
- 设<的个数为 a，=的个数为 b，大于的个数为 c。
- 如果 a=1 或 c=1，无论怎样排列都可以。方案数为 n!/(t_1!t_2!...t_k!)
- 否则 a>1 且 c>1，源点必须在一起，汇点也必须在一起，否则会出现多种不同的答案。方案数为 C(n,b)*a!/(p_1!p_2!...p_x!)*b!/(q_1!q_2!...q_y!)。

5D

- 设 $f(v_0,v,a,l)$表示初速度为 v_0，最大速率为 v，加速度为 a，通过长为 l 的路段的最短时间。
- 通过 $v^2-v_0^2=2ax$ 可以解出直接从 v_0 加速到 v 的距离，如果>=l 说明一直加速，否则是加速完之后按照 v 的速度继续前进，分别计算一下即可。

- 如果 v<=w（最大速率）也不会超速就直接开，直接调用 f。
- 如果不满足则解出速度到达 w 的距离，如果>=d 就直接开，否则分两种情况，一种是先加速后减速，一种是先加速再匀速最后减速。
- 设 t 为加速阶段的时间，则(at)^2=2ax_1,w^2-(at)^2=2(-a)x_2,x_1+x_2=d，解得 t^2=(2ad+w^2)/a^2。
- 如果 at<=v 就按照这么来，否则你算出加速到 v 的时间，之后匀速再减速，到了 d 之后就可以调用 f 走到 l。

1216E2

- 考虑分长度讨论。1..1 的长度为 1，1..2 的长度为 2，...，1..9 的长度为 9.
- 同理 1..10 的长度为 11，1..11 的长度为 13，1..12 的长度为 15，...，1..99 的长度为 189.
- 相当于是若干个等差数列, 1..9 的长度是 9*1, 1..99 的长度是 9*1+90*2, 1..999 的长度是 9*1+99*2+900*3, 以此类推。
- 由于 k<=10^18，容易计算出 1..x，x 的位数。
- 之后用二分或直接降位都可以。

711E

- 不妨改为任意两人生日不同的概率最后用 1 减去。那么概率显然是 A(2^n,k)，而总方案数是 2^{nk}。
- A(2^n,k)=(2^n-k+1)(2^n-k+2)...2^n，k>Mod 显然为 0，否则可以将 2^n 对 Mod 取模，然后暴力。

1155E

- 由于次数不超过 10，可以直接求出 11 个点的点值然后拉格朗日差值。
- 然后暴力枚举所有位置看是不是零点。

1594E2

- 总点数很多，关键点数比较少。对于子树没有特殊点的可以直接算，有特殊点的不超过 n*k 个。
- 没特殊点的记 dp_i 表示深度为 i 的满二叉树染色方案数，算一下即可。
- 有特殊点的直接暴力 dp。

639C

- 你先把绝对值>1 的进/退位，到某一位如果值不为 0，则说明这一位之后要改。(之前的位显然不影响这一位)。
- 得到前面位（包括这一位）的值，如果>k 显然就没救了，否则你再把这个数往下退位，那一位看能不能加上这个值。

925C（#）

- 设当前异或值为 s，那么之后能加的数 v 要满足 s xor v>s，即二进制下 s 在 v 的最高位上的值为 0。
- 可以发现**最高位比较低的并不会影响最高位高的决策**，考虑从大到小加入最高位为 i 的数。
- 现在序列里已经有最高位>i 的数了，要加入最高位为 i 的数（设有 y 个），由于加入之后并不会影响那些大数的大小关系，但你新加进去的数的位置必须满足当前前缀异或起来第 i 位是 0，考虑最高位>i 的数里有多少个第 i 位为 1（设有 x 个），如果 y>x+1 则无解，否则一定可以加入。

601C

- f[i][j]表示前 i 场总分=j 的期望人数，转移的话除了 c[i]其它都是等概率的。
- 那么答案是 f[n][<sum]+1。

1101G

- 如果**所有数的异或和为 0**，选所有子段的异或和一定为 0，**无解**。
- 一个集合的任何子集的异或和非 0，说明这个集合是线性基。
- 你相当于选出若干个位置 a_1,a_2,...,a_k, pre_{a_1}^0,pre_{a_2}^pre_{a_1},...,pre_n^pre_{a_k}, 要是线性基。
- 不妨求个前缀异或和，即 pre_{a_1},pre_{a_2},...,pre_{a_k},pre_n 要是有个线性基。
- 你把 pre_n 加入，剩下的能加入线性基就加入线性基（线性基大小和顺序无关）。

793C

- 考虑算出每只老鼠在捕鼠器内的**时间区间，然后求个交**（初始设为[0,INF]）。
- 两个维度分别处理，即求出 x 在[x1,x2]的时间和 y 在[y1,y2]的时间，求交即在捕鼠器内的时间。
- 对于一维，如果老鼠一直不动，则判断是否在区间内，不在则无解。

- 考虑沿着速度方向走，求出到 x1,x2 的时间，可行时间区间为[较小值，较大值]（小于 0 其实没关系，和外面求交的时候就消失了）。

612E（#）

- 这种看上去就是**对排列建图**。考虑把 q_i 变成 q_{q_i}会怎么样。
- 对于长度为偶数的环 1->2->3->...->2k->1，会拆成 1->3->5->...->2k-1，2->4->6->...->2k 两个长度为 k 的环。
- 对于长度为奇数的环 1->2->3->...->2k+1->1，会变成 1->3->5->...2k+1->2->4->6->...->2k。
- 那么对于 p 长度为奇数的环，换一下位置调整成 q 中的环。p 中两个相同的长度为偶数的环合并成 q 中的环。

1610E

- 先思考如何**快速判断一个不降序列是不是好的**，拿出一个子序列，如果最小值=最大值则显然是好的。
- 如果不相等，如果存在 x,y,z(x<=y,y<=z,x<z)，{x,y,z}如果是可怕的，那么序列显然不是好的。此时 x 一定小于 avg，z 一定大于 avg，是可怕的则 y>(x+y+z)/3，即 2y>x+z。
- 会发现 x 一定取序列的最小值，y,z 会取两个相邻的值。
- 看是否满足这个条件就行了，因为假设是可怕的，>avg 的最小值是 y，那么你剩下的平均值要越小越好，且 <avg 的个数<=除了这个数外>avg 的个数，那只会取 x,z。
- 枚举最小值，那么有 2(y-x)<=(z-x)，每次取的数字至少*2，可以发现能贪心的往后取第一个最小的，而且每次都至少乘 2，所以只会做 log 次，可以直接暴力二分，复杂度两个 log。

993E

- 对于区间[l,r](l<=r)，它会将 S_r-S_{l-1}的值+1。
- 先不考虑 l<=r 的条件，那么这样会导致 0 的值错误。不过 0 的值很容易计算。
- 将 l-1，现在就变成对于任意 0<=l<n,1<=r<=n，求 S_r-S_l。
- 设 f(x)=x^{-S_0}+x^{-S_1}+...+x^{-S_{n-1}},g(x)=x^{S_0}+x^{S_1}+...+x^{S_n}。
- 求出 f*g 的 1..n 的系数就是答案，0 的单独计算即可。
- 但是这样 f 指数是负的不好计算，将 f 的指数都加上 n 即可。

1254C（#）

- 和正常求凸包一样**考虑相邻两个点**。看其他点到这两个点组成直线的距离，逆时针方向上一定是先增后减的。
- 找到那个最远的点，分割成两部分。现在我们想知道哪些在逆时针到最远点之前的半边。考虑从小到大加点，维护一个栈，如果当前点和栈末尾两个点形成的角不是逆时针，如果不是，说明栈末尾的点是假的，弹出。再将当前点加入栈，这样就分成了两部分，需要 2(n-2)步。
- 现在关键在于如何找到两个相邻的点。钦定一个点 i 是 1，另一个点 j 初始是 2，然后枚举剩下的点，看有没有点在 ij 的逆时针方向，如果有就替换。

1505F

- 第一张图是 Nemeth Braille Code for Mathematics，第二张是统一英文盲文（Unified English Braille）。
- 这两张图表示的都是 2-x^2。

830C

- 先转化成整除的形式，变成\sum (-a_i+([(a_i-1)/d]+1)*d)<=k。
- 把 a_i 的和移到等式右边，d*\sum ([(a_i-1)/d]+1)<=k+\sum a_i。
- 直接枚举 d 显然复杂度过高，考虑对于每个 i，(a_i-1)/d 只有 sqrt 种不同取值。
- 把每个分界点加入到序列，排序，更新即可。

906B（#）

- 不妨假设 n<m，有一种想法第二行是第一行+2，第三行是第二行+2…（存的是第 i 行的第几个位置）。
- 若 m>4，第一行有一种显然的**构造是 m,m-2,m-4,…，然后再 m-1,m-3,m-5,…**。m=5 为 53142，m=6 为 642531，它们无论加上什么都会保持相邻绝对值差>1。
- **小范围可以暴搜。**

772C（#）

- 考虑当前的前缀积为 a，你要变成 b，需要填一个 x 满足 ax mod m=b，转化为 ax+my=b。
- 如果 gcd(a,m)不是 b 的约数则无解，否则存在方案。即 a 能到所有 gcd(a,m)的倍数。
- 没必要暴力连边，gcd(a,m)也只有 m 种取值，连一条 a,gcd(a,m)的 0 边，gcd(a,m)再连到它的倍数。
- 这样边数是 O(m log m)的，再用 0/1 最长路求出前缀积，还原序列可以用 exgcd 实现。

113C

- 根据费马平方和定理，奇素数能表示为两个平方数之和的充分必要条件是该素数被 4 除余 1。
- 所以求的是区间 4k+1 型的素数个数（如果区间包含 2 答案要+1），直接欧拉筛即可。

938E

- 可以发现，值等于最大值的数一定不可能有贡献。
- 对于不是最大值的数，它有贡献当且仅当排在它之前的数都比它小。
- 设有 k 个比它小，那么答案是 C(n,k)*k!*(n-k-1)!。（第一个非比它小的位置放当前数，比它小的和比它大的内部任意）。

441D（#）

- **根据经典套路，先对排列建图。**根据经典结论，f(p)=n-环数（每个环需要的次数是环长-1）。
- f(p)<m 说明要减少环的数量，选两个环上的两个元素交换一下，就把环接起来了，数量-1。
- f(p)>m 说明要增加环的数量，选一个环交换两个。
- 操作的时候按照字典序最小的来即可。

1146D

- ua+vb=gcd(a,b)是一定有解的，且一定有 u>=0,v<=0 的解（否则可以调整）。
- 但是不一定在范围内，不过可以发现 x>=a+b 的时候一定可行，因为假设 a>b，可以每次尽量减 b，如果减不了说明当前<b，可以增加 a（<a+b）。
- 否则如果<a+b，直接暴力最短路计算到达当前点需要经过的最大值是多少，将最大值不超过 x 的加入 f(x)。

1156F

- 有了 x>y 才继续的限制，说明抽出来的卡一定是越来越小的，直到最后一步失败和胜利。
- 那么可以直接 dp，设 dp[i][j]表示抽了 i 张当前抽到了 j 的概率，那么从\sum_{k<j} dp[i-1][k]*tot_i/(n-i+1)转移过来。
- 发现 k 和后面乘的东西无关，记录一下 dp[i][*]的前缀和即可，预处理逆元可以做到 O(n^2)。

845G

- 你可以选择一条路径，再异或上若干个环的权值，即可表出所有路径的权值。
- 可以考虑证明，先取一棵 dfs 生成树（1,n 不连通显然无解）。
- 然后你加非树边，所有的环经过这两点的环都可以通过这个一条非树边和其它树边形成的环和其它的环异或出来。(你考虑它的两端点，有多条路径说明是多个环异或出来的)。
- 所以就取生成树，然后加入生成树上形成的环，再找一条路径。现在看的就是路径和这些环能组成的最大值。
- 那么就可以建线性基然后贪心了。

895D

- 相当于给定字符集，求字典序在>a 且<b 中的方案数。
- 如何计算？枚举 lcp 长度，再枚举 lcp+1 位置的字符，看是否可行，用组合数计算剩余字符填的方案数。
- 复杂度为 O(n|Σ|)。

615E

- 除了第 1 层之外，第 k 层的点数为 6(k-1)。
- 根据等差数列求和，1..k 层的点数为(0+6k-6)k/2+1=3k^2-3k+1。
- 算出层数后再算一下在哪个位置，按照左上，左，左下，右下，右，右上的顺序，每个都是 k-1 步，看一下在哪个阶段即可。

300E

- 由于是阶乘乘起来，可以通过后缀和计算每个数具体被乘了几次。
- 然后要计算每个数中每个素因子出现了几次，可以直接暴力 p^a，然后再乘 k（k*p^a），计算数量。
- 之后可以二分然后判断是否可行，也是枚举 p^a 看有多少个，判断是否<=原式中的 p 的数量。

898F

- 两个数加起来等于第三个数，那么第三个数的位数要么是加数位数的较大值，要么是较大值+1。
- 双哈希判断一下，如果合法，也可以暴力判断保证一定是对的。

818F

- 首先一棵树一定满足条件，接下来加一条边会使至少两个点在同一个边双。
- 有没有可能最后有多个边双呢？答案是否定的，因为有多个不如一个，能加的边数和边双大小呈二次函数关系，所以一定只有一个边双。
- 设这个边双有 x 个点，那么边数应该为 min(C(x,2)+n-x,2(n-x))（因为得满足割边数量超过一半）。
- 左边的式子=x*(x-3)/2+n，右边的式子是-2x+2n，左边递增右边递减，求一个左边小于等于右边的最大的 x，判断 x 和 x+1 即可。

773C（#）

- 最后答案一定是一个区间，考虑如果求出了最小值，那么把那些 r=1 的位置的 r 变成上一个数，在把 1 单独当一个数，答案就+1 了，最大值是 1 的个数。
- 记 a_i 表示 2^i 的个数，b_i 表示(2^i,2^(i+1))中的数的个数。
- 考虑二分最小值，具体判断方法是，从小到大枚举 i，记录还能在后面继续加的数的数量 x 和之前剩余的可以当 r 的个数 y，尽量加 a_i，如果 a_i<=x，则用完 a_i，剩下的尽量用 y，x 变成 a_i；否则将 a_i 多余的给 y，y+=a_i-x。

268E

- 显然所有歌都会至少听过一遍。
- 考虑只有两首歌咋做，两种情况，a,b 的顺序期望多听 p_a(1-p_b)l_a，b,a 的顺序期望多听 p_b(1-p_a)l_b.
- 对于更多首歌的情况，可以调整，考虑相邻两首歌，那么之前的和之后的和它们的相对顺序都没有影响，如果满足 p_a(1-p_b)l_a<p_b(1-p_a)l_b 的条件，交换即可。
- 这个式子移项就变成 p_al_a/(1-p_a)<p_bl_b/(1-p_b)，可以直接 sort。

768F

- 直接枚举酒有 x 堆，那么合法方案数是 C(w-x*h-1,x-1)，所有方案数是 C(w-1,x-1)，食物的堆数可能是 x-1,x,x+1，分别对应方案数 C(f-1,x-2)，2*C(f-1,x-1)，C(f-1,x)。
- 两种相乘然后相加可以算出总合法方案数和总方案数。

460D（#）

- k=1 的话选 l，r<l+4 可以暴力。
- 令 u=l+(l&1)，k>=4 可以选 u,u+1,u+2,u+3（答案为 0），k=2 可以选 u,u+1（答案为 1）。
- k=3 要么用 k=2 的构造方法，要么构造 a>b>c 异或值为 0。
- 枚举 a 的最高位 k，a,b 都会有 2^k，c 没有。
- 枚举 c 的最高位 i，那么 a 只会有 c 的最高位（尽量小），剩下的位给 b，c 最优是(2^i)-1（尽量大）。

518E

- 把式子重复项删去，变成 a_1<a_{k+1},a_2<a_{k+2},...,a_{n-k}<a_k。
- 即 a_1<a_{k+1}<a_{2k+1}<...，a_2<a_{k+2}<a_{2k+2}<...，...，a_k<a_{2k}<a_{3k}<...。
- 这个要求绝对值之和尽量小，就不能用经典套路了。不过还是可以对两个固定位置之间的取值分类讨论。
- 如果两个固定的都是负的，中间的数尽量往大取。如果两个固定的数都是正的，中间的数尽量往小取。否则要使得绝对值之和尽量小，中间的数正负的个数尽量平均。
- 为了方便起见，可以认为每个这样的下标差为 k 的序列中，开头有个负无穷，结尾有个正无穷。

336D

- 因为任何数和 1 搭配都会变成 0，所以从左往右第一个 1 的位置之后的都可以不管（一定会变成 1）.
- 对于 0000...01，做一次操作会删去最后的 1；对于 0000...00，做一次操作会将最后的 00 变成 1.
- 所以如果是奇数个 0 一个 1，最后会变成 0；如果是偶数个 0 一个 1，最后会变成 1.
- 直接枚举第一个 1 的位置，组合数计算一下后面的方案数。

915G

- 固定 i，方案数为\sum_d mu(d)[i/d]^n。在 i 增加的时候，[i/d]≠[(i-1)/d]说明 i%d=0。
- 所以只需要枚举 i 的约数，计算新增的贡献即可。(对于每个数的 n 次，可以先预处理)

255E

- sg 打表，从四次根号到二次根号里求个 mex。
- x 比较小直接输出 sg 值。
- x 比较大，开个根号就足够小了，预处理前缀和计算 sg 为 j 数量，就可以求 mex 了。
- 可以发现 sg 值不大，所以可以直接暴力。

- 有另外一种方法是直接打表，sg 值是整段出现的。
- <4 的值为 0，<16 的值为 1，<82 的值为 2，<6724 的值为 0，<50626 的值为 3，<2562991876 的值为 1，剩下的值为 2。(大概也能看出来是成段出现的，4 次根号和 2 次根号跨度很大，当 n 很大的时候下界都差不多)。

217B

- 直接暴力枚举最后达到条件后另一个数是什么，然后暴力，每次大减小，得到下一次的值。
- 如果最后是 0,1，且第一步是 T，步数比当前最优方案小就替换。
- 暴力复杂度是 O(nr)，其实可以先用辗转相除算出步数，再暴力输出方案。

1200F

- n 很小，可以考虑拆点。但是 i 不能只拆成 m_i 个点，因为其它点取模的数字不同，需要 mod 它们的 lcm，即

lcm(1,2,...,10)=2520。

- 现在每个点只有唯一出边，可以直接找环。询问时可以直接输出。

660E

- 他说要包含空序列，先算上空序列的个数 m^n。
- 对于非空子序列，常见的表示方法是拿出在**原序列第一次出现的位置**。
- 考虑枚举子序列长度 i，有 m^i 种方案，再枚举子序列最后一个对应的位置 j，有 C(j-1,i-1)种选法，之前的 j-i 个未确定的位置只能有 m-1 种方案，最后的 n-j 个位置任意。
- 把 j 提到外面，令 i,j 都减 1，里面就变成二项式定理的形式，最后变成枚举 j(0<=j<=n-1)，m^{n-j}*(2m-1)^j。

1039A（#）

- 题目给定 a_i 是单调递增的，那么 a_i+t 也是单调递增的，且 b_i 是单调递增的，所以 b_i>=a_i+t。
- **那么说明 x_i>=i，否则一定无解**。否则若 x_i=c，那么删除 b_c 和 a_i 后对应位置依旧需满足 b>=a+t。即变成 b_i>=a_{i+1}+t,b_{i+1}>=a_{i+2}+t,⋯,b_{c-1}>=a_c+t。**满足条件的情况下要求 b 尽量小**（否则 x_i 可能>c）。
- 最后再判断一下 x_i 是否=c 即可。

509D（#）

- 这么经典的题为什么过的人这么少。。。
- 考虑**如果得到了一组解 a,b，那么 a-x,b+x 也是一组解**。
- 那么**不妨假设 a[1]=0**，那么 b[i]=a[1][i]，同理可以解出 a[i]=v[i][1]-b[i]。
- 现在要所有(a[i]+b[j]-v[i][j])%k=0，对所有|a[i]+b[j]-v[i][j]|求 gcd。
- 如果 gcd<=矩阵 max，那么无解，否则取 k=gcd 即可（注意特判 gcd=0）。

126D

- 根据经典结论，每次贪心的选最大的不超过 n 的数，就可以求得一种方案。
- 这种方案显然是选出的数最大的，且方案中不存在两个相邻的数都被选。
- 现在要类似于退位，就是把 1 摊到后面的位。可以发现如果有相邻的两个 1，后面的 1 是无论如何都拆不了的。
- 那么从小往大考虑，如果当前数不拆，那么显然和上一个数无关；如果当前数拆，知道了上一个 1 拆/不拆，就可以知道这一段区间有多少个 0，方案数只和中间 0 的个数有关（000...0011，如果拆变成 000...1101，后面的 01 可以忽略，那么总长度减少 2），方案数应为 0 的个数/2 下取整+1。
- 设 dp[i][0/1]表示到第 i 个 1，当前这个数拆/不拆的方案数，从 dp[i-1]转移。

225E

- **分类讨论 x 是奇数还是偶数**。
- x 是奇数将 x 替换成 2x+1(x>=0)，z=x+2y+2xy，即 z+1=(x+1)(2y+1)，说明 z+1=2^k(k>=0)>
- x 是偶数将 x 替换成 2x(x>=1)，z=x+y+2xy，即 2z+1=(2x+1)(2y+1)，说明 2z+1 只能是偶数或奇素数或 1（2z+1 显然不是偶数）。
- z=1 时可行，特判 n=1。
- 其它情况要求 2z+1=2^{k+1}-1(k>=0)且是奇素数，这个显然是奇数，所以问题就变为求第(n-1)个梅森素数，

OEIS 有 A000668 为第 n 个梅森素数，不过你可以用指数的表 A000043，更容易存储。

1599H

- 询问一下网格左上和右上，那么询问中间的距离应该是先-1，再 0，最后加 1 的折线形式。
- 不妨设左边的距离较大，则可以求出和右边距离一样大的点，它和右边的中点就是横线的中点，一定是最小值。
- 你求出了矩形左上和右上的横纵坐标，接下来同样询问纵轴（左上和左下），但是可以发现左上没必要再问一次，所以总次数是 5 次。

954I

- 把 a 变成 b，相当于把 a,b 看成同一种。枚举字符的等价类（集合划分）。
- 然后就可以类似 kmp 计算答案了，时间复杂度 O(Bell(|Σ|)*n)。

366E

- 求相邻的 a_{i-1},a_i 在矩阵中曼哈顿距离的最大值的最大值。
- |x1-x2|+|y1-y2|就是 4 种差的 max，分别记录一下最大值就可以计算了。
- 以 x1-x2-y1+y2 为例，x-y,-x+y 的最大值相加就是这种情况。

1425D

- (B_1+B_2+...+B_k)^2=\sum_i B_i^2-2\sum_i\sum_j [i≠j]B_iB_j。
- 要计算包含 i 的方案数和同时包含 i,j 的方案数。包含的方案数就用总共减去不包含的方案数。
- 有多少方案同时包含 i,j，可以用两个都不包含的方案数+包含 i 的方案数+包含 j 的方案数-总方案数计算。
- 它这个距离是切比雪夫，所以用二维前缀和还是比较容易计算的。在不包含里选 m 个即可。

730D

- 可以贪心，走完第 i 座桥，首先要求喝的饮料数尽量少，其次要求当前饮料剩下来的时间尽量多。
- 新来了一座桥，如果能直接走过去不超时就直接走，否则先把当前剩下的双倍时间走完，然后看能不能直接走（t>=2l）到下一个，能就走，否则先把能走的 t-l 暴力走（l-x<=t-2x,x<=t-l），接下来必须要一直喝饮料，只要>=r 就开始暴力喝饮料，距离-r（如果 ans 过大直接用除和模代替）。

596D

- 你考虑设 dp(l,r,x,y)表示只考虑 l,r 内的树，左边向左倒/向右倒，右边向左倒/向右倒。
- 如果左端点往右倒，考虑倒了带了几个，转移；如果右端点往左倒，考虑倒了带了几个。
- 倒了带几个是固定的，预处理一下即可，注意如果倒了会带上整个区间需要特判。

115C

- 如果一行里有固定的，那么这一行每个位置是朝左还是朝右就确定了，相邻的两个方向不同。
- 如果一列里有固定的，那么这一行每个位置是朝上还是朝下就确定了，相邻的两个方向不同。
- 如果存在一行/一列有矛盾则无解。否则一行没固定的方案有 2 种，否则方案有 1 种，列同理，乘起来即可。

150C

- 在[c,d]逃票的期望就是 c 到 c+1，c+1 到 c+2，...，d-1 到 d 的期望和，y 到 y+1 的期望是 x_{y+1}-x_y-p*c。
- 记 y 到 y+1 的期望为 val[y]，可以发现，询问[a,b]即询问 val 数组[a,b-1]中的最大子段和。
- 线段树维护即可（维护前缀最大，后缀最大，最大，总和）。

167C

- f(a,0)=0,f(a,1)=1。要求 f(a,b)（a>=b），先考虑 f(b,a mod b)，如果是必败那显然直接取。
- 如果是必胜，那么考虑做 f(a-b^k,b)，但是你发现还得 a-b^k>=b，否则相当于变成 a mod b 了，而且你无论减多少次 b^k，a mod b 的值还是不变。
- 将 a-b^k 看成 a-(b^{k-1})b，那么不能减到 a-[a/b]b。所以问题变成，初始是 a/b，每次可以取 b^0,b^1,...，取完的人失败。
- 这个东西和巴什博弈（减法博弈）比较像，考虑和那个一样每次无论对方做什么操作，都可以把它移到一个必胜态。考虑在 mod (b+1)下，只有取 1 和 b 两种，你可以控制它 mod (b+1)的值不变，同理对面也一样。最后到<b+1 的，胜负就和奇偶性有关（奇败偶胜）。
- 如果 mod (b+1)是奇数，那么后手直接和你反着来，你去 mod (b+1)为 1 的他取 b，反之亦然，你就输了。否则如果是偶数，如果当前数 mod (b+1)不为 0，你直接取 1，就能使它 mod (b+1)的值减少 1（变成奇数），否则取 b，使得它 mod (b+1)的值为 1（奇数）。
- 对于原问题，就递归看 f(b,a mod b)是否必败，如果必败那么当前状态必胜，否则就按照上述方法计算。

385E

- 把坐标搞到 0..n-1。

$$
\begin{aligned}
dx_i &= dx_{i-1} + sx_{i-1} + sy_{i-1} + (i-1) + 2 \\
dy_i &= dy_{i-1} + sx_{i-1} + sy_{i-1} + (i-1) + 2 \\
sx_i &= sx_{i-1} + dx_i = 2 \times sx_{i-1} + dx_{i-1} + sy_{i-1} + (i-1) + 2, \\
sy_i &= sy_{i-1} + dy_i = 2 \times sy_{i-1} + dy_{i-1} + sx_{i-1} + (i-1) + 2
\end{aligned}
$$

- 多记个 i 和 1，6*6 的矩阵快速幂即可。

1387A

- 不同连通块是独立的，考虑一个连通块。
- 假设一个点的点权是 x，那么可以求出其它点和 x 的关系式（也有可能无解或解出 x）。
- 如果无解直接输出-1，解出 x 可以直接计算贡献。
- 否则就是若干个|x-v_i|之和，要求它最小。经典小学数学，数轴上有若干个点，要离所有点距离之和最小，根据结论，在中位数取最小（两个中位数可以在它们之间任何一个点上），解出 x 可以求连通块所有点的值。

1045D

- 根据经典套路，**计数连通块个数可以计算连通块的最高点的数量**。对于这道题而言，可以计数父亲不存在，而自己存在的点的期望，\sum_u (1-p[fa[u]])p[u]。
- 考虑修改，会对所有儿子和自己的贡献产生影响，但是不能直接 for 所有儿子。
- 不妨记录 sum[u]表示 u 的儿子的 p 值之和，那么修改的增量可以快速计算，且 sum 也在修改后容易维护。

316E3

- 这种题看上去就很线段树。但是线段树需要快速合并。
- 区间[l,r]的线段树维护的是询问[l,r]的式子（l 对应 x=0），但是不是很好合并。[l,mid],[mid+1,r]，mid+1 里的 f 的下标都要增加 mid+1-l。其实也不难，因为系数是斐波那契数列，不妨再维护一个 l 对应 x=1 的，那么 l 对应 x=2,3,4,...都可以由前两个加起来。
- 具体的，设 s_i 表示第一项对应 x=i 的，那么 s_i=s_{i-1}+s_{i-2}。这依旧是斐波那契递推式，它们的系数是一般的斐波那契，s_i=f_{i-2}s_0+f_{i-1}s_1。
- 那么就可以合并了，大区间的 s_0=左区间的 s_0+右区间的 s_{lenl}，大区间的 s_1=左区间的 s_1+右区间的 s_{lenl+1}。询问就直接询问。
- 单点修改操作可以直接做。区间加的话需要打标记，考虑如何快速求出加之后当前点的权值，可以发现加上了(f_0+f_1+...+f_{len-1})d=(f_{len+1}-1)d，可以快速计算。

852F

- 把序列翻过来，变成 A[i]*A[i-1]，下标从 0 开始。
- 写成一个二维矩阵的形式，第 i 行表示 i 次操作后的 A[j]是 a 的几次方（i>=0）。

```
1
1 2
1 3 4
1 4 7 8
1 5 11 15 16
1 6 16 26 31 32
```

- 把第 i 行前 i 列拿出来，形如···
- 除了最后的 1,2,4,8,16,...，其它的都是按照组合数形式递推的。
- 但是对于 1,2,4,8,16，它们左边的位置正好是它-1。
- 令第 j 列的值都减去第 j-1 列的值，就变成 1 了，符合递推公式。

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
···
```

就是普通组合数。

- 所以原来对应的就是组合数前缀和。计算一下，由于 phi 的值很小，可以直接对 phi 取模。

305D

- **只会有跨距离为 k+1 的边**，否则如果两个点的最短路跨了多条路径，那么跨一条边之后，它到当前终点的最短路径就不对了。当然还有 i 到 i+1 的普通边。
- 所以，不存在一条路径跨了两条距离为 k+1 的边，所以距离为 k+1 的边的开头的差值不能超过 k。
- 所以你判断一下读入的，是不是只有普通边和跨距离为 k+1 的边，然后看跨距离为 k+1 的边的开头的最小值 L 和最大值 R，如果 R-L>k 就无解，否则，你可以再加若干条边，要求开头的差值不能超过 k。
- 那么直接枚举选择的最右边的一条边的开头是什么（注意+(k+1)后不能超过 n)，其它的方案数用 2 的幂次算一下即可。

271E（#）

- 这道题是不是也属于比较阴间的找不变量？**考虑所有卡片 y-x 的差值。**
- 1 操作出现一张差值不变的卡，2 操作出现差值/2 的卡，3 操作出现两个差值相加的卡。
- 设 d=y-x，那么不停/2，得到 d'(奇数),d=d'*2^k，那么无论哪一步产生的都是 d'的倍数。
- 猜测只要 d'|gcd(a_1-1,a_2-1,…,a_n-1)就可以了，证明：
- 定义 13 操作：(x,y)通过一操作可以获得(y,2y-x)，再通过三操作获得(x,2y-x).
    1. 必要性：你每次只能得到 d'的倍数，如果 gcd 不是 d'的倍数，说明有数不是 d'的倍数，凑不出来。
    2. 充分性：开始是(x, x+d'*2^k)，通过一次 13 操作可以变成(x, x+d'*2^{k+1})，通过若干次能变成(x, x+d'*2^{100})，再利用 1 操作变成(2^{100}, (1+d')*2^{100})，再不断使用 2 操作变成(1,1+d')，之后可以通过 13 操作变成(1,1+2d'),(1,1+3d'),…，由于 d'|a_i-1，所以一定能表出所有数。
- 枚举 gcd 的奇约数，枚举它的 2^k 倍即可得到 d，即 y-x 之差，计算方案数即可。

491B

- 把曼哈顿距离的绝对值拆了，分别记录 x+y,x-y,-x+y,-x-y 的最大值。
- 对于每种方案，算一下四个的最大值是否比当前答案小，小就替换。

1575K

- 如果 a=b，相当于没有限制，方案数是 k^{nm}。
- 如果 a,b 两个矩形不相交，方案数是 k^{nm-rc}。
- 否则如果相交但不完全相同，你把第二个矩形之外的所有格子都填了，方案数是 k^{nm-rc}，然后第二个块和第一个块对应的位置也固定了，此时一定新填了 1 中的格子，继续递归下去，直到填满。所以方案数也是 k^{nm-rc}（可以分覆盖一个角和覆盖两个角的情况讨论一下）。

802D

- Poisson 分布，均值为λ，方差为λ。
- Uniform 分布，均值为λ，方差为λ^2/3（如果下界是 a 上界是 b，那么均值是(a+b)/2，方差是(a-b)^2/12，这里取 a=0,b=2λ）。
- 由于λ>=10，不妨取 3*方差/(均值^2)，那 Poisson 分布期望为 3/λ，Uniform 分布期望为 1，将界定为 0.6 即可。

1575G

- 使用经典 phi*1=id，拆掉 gcd(i,j)变成\sum_d phi(d)\sum_{d|i}\sum_{d|j} gcd(a_i,a_j)。
- 把另一个 gcd 也拆了，变成\sum_d phi(d)\sum_g phi(g) (\sum_{d|i} [g|a_i])。
- 一种牛逼方法是枚举 d 和 d 的倍数 i，再枚举 a_i 的约数 g，这样的复杂度是 O(n ln n d(n))，d 是 1..n 最多有多少个约数，d(1e5)=128；另一种是枚举 d 然后枚举 i,j。
- 把两个做法合起来即可，类似根号分治。

86B（#）

- 从上到下，从左到右，先用 1*2 和 2*1 的来覆盖。剩下的空位找四连通的已经被覆盖的接在一起。
- 如果没有说明四连通都是障碍物，无解。可以证明每个块大小不超过 5。

316E2

- 同 316E3。

212C

- 题目是环形字符串每次将所有 AB 变成 BA，求操作前可能的排列数。
- 考虑破坏为链，复制到最后一个字符上同时钦定这两个字符必须相同。
- 大力 dp，设 dp(i,0/1)表示决策到了第 i 位，填了 A/B 这个字符时合法的字符串方案数。
- 转移，如果填个 A 就没啥关系，它不可能和前面一个翻；如果填个 B，上一个填 A 的话就要翻，否则就不翻。
- 要求翻之前和原来是一样的，注意第一个和最后一个字符相同。

42D

- 由于路径一定是经过每个点一次，**那么设边的权值为 a[u]+a[v]，那么路径权值就固定**为 2*suma。
- 现在要给每个点分配这样的 a，使得所有边权值不同。
- 不妨考虑依次加入 1,2,...,n，那么不存在 a[u]+a[v]=a[i]+a[j](u≠v)，将之前所有 a[u]+a[v]-a[j]加入 set，这些不能选，然后尽量选最小的不在 set 中的正整数即可。
- 然后发现在 n=20 的时候 a[20]=413，符合条件。

37D

- 显然第一次上课，标号随便标。
- 第二次上课，设 dp[i][j]表示考虑完前 i 间教室，有 j 个组没有分配第二个教室的方案数，转移用组合数选一下。

180B

- 每次取 d 和 b 的 gcd，将 d 除以这个 gcd，将 ans 增加 1，直到 d,b 互质。
- 现在如果 d=1，说明 b^ans 是 d 的倍数，可以直接用 2-type。
- 如果不满足，如果 ans=0，说明本来 b,d 就互质，那么看 b%d 是否是 1 或-1，分别对应 3-type 和 11-type。

- 否则现在已经被除掉 d 的中的数可以在最后若干位用 2-type，且和剩下的 d 互质，所以现在只需要考虑剩下的 d 能否用 3-type 和 11-type 来判断，它们分别能判断(b-1)%d=0 的，(b+1)%d=0 的，d 显然是可以质因数分解然后 CRT 的，每个质数的幂次是看 b-1,b+1 对应的质数幂次哪个大用哪个判断，对应的就是 lcm(b-1,b+1)。所以就是如果 lcm(b-1,b+1)%d=0 可以用 6-type。
- 剩下只能 7-type。

50E

- 解方程，变成 x=-b±sqrt(b^2-c)。
- 分类讨论，如果 b^2-c 开根是整数，那么只能是 x,x+1,...,y（x=ceil(sqrt(max(b^2-m,0))),y=floor(sqrt(b^2-1))），区间给[-b-y,-b-x]+1,[-b+x,-b+y]+1（注意如果 x=0 那么-b 位置只能加 1 次）；如果 b^2-c 开根不是整数，那么将不是整数的方案数*2 累加到答案。
- 最后加上位置不为 0 的个数，这个可以排序来计算。

209B（#）

- 先排序使得 a<=b<=c。在没结束时，你要让一个数至少为 2 的数-2，可以用两次（设自己为 x，另外某个非 0 数为 y，剩下的为 z）：x-1,y-1,z+1; x-1,y+1,z-1。
- 如果 a,b 奇偶性相同，至少操作 b 次。先 b-2 直到 a=b，然后再 a-1,b-1,c+1 直到 a,b 都为 0，操作次数为 b。
- 如果 a,b 奇偶性不同，那么无论怎么操作 a,b 的奇偶性都不同，那么只能 c 最后为 0，至少操作 c 次。先 c-2 直到 a=c 或 b=c，然后 a-1,b+1,c-1 或 a+1,b-1,c-1 直到 a,b 或 a,c 都为 0，操作次数为 c。

1600F

- 根据 ramsey 数，R(5,5)在 43 到 48 之间。
- 那么你在前 48 个人里找 5 个即可，直接 dfs。

593C（#）

- **考虑类似 CRT 一样，你构造若干个函数 f_i(z)，在 z=i 的时候返回 x_i，在 z≠i 的时候返回 0，把它们加起来**就可以得到 f(z)了，对于 g 同理。
- 考虑找到一个函数[z=0]，这个可以通过(2-||z-1|-|z+1||)/2 表示，但是题目没有除法，我们只能有 2[z=0]。
- 那么 2[z=i]*(x_i/2)也一样，即 2[z-i=0] *(x_i/2)，可以通过(2-||z-i-1|-|z-i+1||)*(x_i/2)表示，你可以表示出[z=i]x_i（偶数）或[z=i](x_i-1)（奇数），因为除以 2 会取下整。
- 不过问题不大，因为 r>=2，即使圆心是(x,y)，(x-1,y),(x,y-1),(x-1,y-1)也一定在圆内。直接这么做就可以了。

802E

- 前面同 802D。
- 理论上你算均值就可以了，但是 Uniform 分布的误差可能超过可接受的范围。
- 如果是 Uniform，就算最大和最小的均值即可。