中国国家集训队第一次作业

福州一中 董克凡

Contents

Part I
传统题

1 BTREE

1.1 试题来源

Codechef OCT 14 Union on Tree

1.2 题目大意

给定一棵N个点的树，边权均为1，现在有Q组询问，每组询问给出k_i个点$u_{k_i,j}$以及每个点的“管辖范围”$range_u$，当且仅当一个点落在至少一个点的管辖范围内时，这个点称作被管辖的。对于每一组询问，求出被管辖的点的个数。

1.3 算法讨论

我们来考虑$k = 1$的情况。这时只有一个询问点。

对于一个点，我们要求出距离这个点距离d的点有多少个，这是一个经典问题，可以用树分治来解决。我们首先把整个树分治的形态记录下来，对于每一层选择的重心，同时维护到这个点距离小于等于i的数组$sum[i]$，那么对于一个询问点u，就可以向上找到每次的重心gra，然后对$sum[d - dist(u, gra)]$求和。这里会造成重复计算，所以每次还要减掉与u在同一个子树内的情况，同样用一个前缀和数组维护即可。

这一部分单次询问复杂度为$\mathcal{O}(log_2(n))$

当$k \neq 1$时，需要将整个树划分为多个部分，对于每一个部分分别求出答案然后相加。这里定义每一个部分的“中心”为一个输入的管辖点。对于每一个点p，它应该属于$range_x - dist(p, x)$最大的那一个中心x，这样定义的好处就是每一个点的管辖范围是一个联通块，而且一个点被管辖当且仅当这个点被所属的中心管辖。

由于不能对每一个点求出其所属中心，所以应该利用每个点的管辖范围连通这一个性质。对于输入的点，构造一棵“虚树“，虚树包含所有输入点，并且总点数不超过$2 * k_i$。那么这棵虚树上的点，我们求出每个点所属的中心(这里可以使用Dijkstra算法)。然后对于虚树上两个相邻的属于不同中心的节点，可以求出它们之间的边上的分界点。因为虚树上两个相邻节点一定有祖先关系，假设这两个点分别为u, v，不妨设u为v的祖先，定义$range_u$为$range_x - dist(u, x), x \in u_{k_i,j}$的最大值。那么分界点$p$应满足：

$$\begin{aligned} range[u] - dist(u, p) &\leq range[v] - dist(v, p) \\ range[u] + depth[u] - depth[p] &\leq range[v] - depth[v] + depth[p] \\ 2 * depth[p] &\geq range[u] - range[v] + depth[u] + depth[v] \end{aligned}$$

取满足条件的$depth[p]$最小的p即可。

这一部分，我们将整棵树划分成多个部分，每一个部分是连通的，多个部分之间互不相关，每个部分由一些分界点确定，分界点总和不超过k_i。此处复杂度为$\mathcal{O}(k_i * log_2(n))$

现在对于每一个部分U，只需要求出这个部分的中心c_U能管辖到的点的个数，然后所有部分求和就是答案了。

现在对于一条边x, y，设$x \in U, y \notin U$，z为分界点。那么我们需要考虑两种情况。

1.当z为x的祖先的时候，我们直接加上在z的子树内，距离c_U不超过$range[c_U]$的点的个数。

2.当x为z的祖先的时候，就减去在在z的子树内，距离c_U不超过$range[c_U]$的点的个数。

这样，就可以求出这一个部分内的答案了，由于每一个联通块一定可以表示为一个子树减去多个子树的形式，所以这一部分的计算是不会有重复或遗漏的。接下来我们考虑在子树z内，距离x不超过$range[x]$的点的个数。这里也要分两种情况讨论。

1.当z不是x的祖先的时候，那么答案就是在z的子树内，距离z不超过$range[x] - dist(x, z)$的点的个数。这一部分可以使用DFS序+可持久化线段树完成。

2.当z是x的祖先的时候，首先答案先加上距离x不超过$range[x]$的点的个数，这里使用$k = 1$的算法即可。然后应该减去那些不在z的子树内的点。也即距离z不超过$range[x] - dist(z, x)$的点的个数。这里多减去了一部分在z的子树内的点，所以需要再加上在z的子树内，距离z不超过$range[x] - dist(x, z)$的点。

至此，我们可以求出每一部分的答案，然后求和就能得到总答案。

1.4 时空复杂度

时间复杂度：$\mathcal{O}(\sum k_i * logN)$

空间复杂度：$\mathcal{O}(N * logN)$

2 QPOLYSUM

2.1 试题来源

codechef DEC 12 Quasi-Polynomial Sum

2.2 题目大意

给定一个多项式$P(x) = c_d * x^d + ... + c_1 * x + c_0$，给定非负整数$q$，正整数$m, n$，求$S(n) = (P(0) * q^0 + P(1) * q^1 + ... + P(n-1) * q^{n-1})\%m$的值。多项式以$[0, d]$之间整数的点值形式给出。

$n \leq 10^{10^5}, m \leq 10^{18}, d \leq 20000$，$m$不能被2至$d + 14$中的任意一个数整除。

2.3 算法讨论

首先，有一个前置技能是，知道一个n次多项式的前$n + 1$项，可以在$\mathcal{O}(n)$的时间内求出其任意一项。[1]

观察小数据可以发现，$S(n) = f(n)q^n - f(0)$，其中$f()$为一个不超过$d + 1$次的多项式。考虑归纳证明：

当$d = 0$时，根据等比数列求和公式，得$S(n) = \frac{1-q^n}{1-q} * c_0$，那么$f(x) = 1/(q-1)$。

当$d \neq 0$时，作$(q-1)S(n)$

$$\begin{aligned} qS(n) - S(n) &= \sum_{i=0}^{n-1} q^{i+1}P(i) - \sum_{i=0}^{n-1} q^i P(i) \\ &= \sum_{i=1}^{n-1} q^i P(i-1) + q^n P(n-1) - \sum_{i=1}^{n-1} q^i P(i) - P(0) \\ &= \sum_{i=1}^{n-1} q^i \left(P(i-1) - P(i)\right) + q^n P(n-1) - P(0) \end{aligned}$$

其中$P(i-1) - P(i)$是度数不超过$d-1$的多项式，根据归纳假设，和式第一项可以写成$q^{n-1}f(n-1) - f(0)$，故$f'(n) = f(n-1)/q + P(n-1)$。

注意以上推导仅对$q, q-1$均有逆元时成立。那么不妨设$m = m_1 * m_2 * m_3$，其中$(m_2 * m_3, q) = 1, (m_1, m_3, q-1) = 1$，那么对于模$m_1, m_2, m_3$分别求出结果以后，即可用中国剩余定理合并得到模$m$的结果。

1.模m_1的做法。不妨设$q^u = 0(\%m_1)$，其中u是最小的满足条件的正整数。那么$S(n)$中不为零的项不会超过$log_u m_1$项，直接暴力即可。

[1] http://blog.miskcoo.com/2015/08/special-polynomial-linear-interpolation

2.模m_2的做法。不妨设$q^v = 0(\%m_2)$，其中v是最小的满足条件的正整数。注意到

$$\begin{aligned} S(n) = \sum_{i=0}^{n-1} q^i P(i) &= \sum_{i=0}^{n-1} (q-1+1)^i P(i) \\ &= \sum_{i=0}^{n-1} \sum_{j=0}^{i} \binom{i}{j} (q-1)^j P(i) \\ &= \sum_{j=0}^{v-1} (q-1)^j * \sum_{i=0}^{n-1} \binom{i}{j} * P(i) \end{aligned}$$

由于$\binom{i}{j}$是一个关于i的j次多项式，故$\binom{i}{j} * P(i)$是一个$d+j$次多项式，那么该多项式的前缀和是一个不超过$d+v$次多项式。然后直接暴力即可。

3.模m_3的做法。首先要求出f数组，注意到由于f是一个d次多项式，故$\sum_{i=0}^{d+1} \binom{d+1}{i} * f(i) = 0$，因为$S(n) = f(n)q^n - f(0)$，所以$f(i), i > 0$均可用$f(0)$现行表示，这样就得到了一个关于$f(0)$的线性方程，求解即可得出$f$数组。然后套用公式即可推出结果。

2.4 时空复杂度

时间复杂度：$\mathcal{O}(D + logN)$
空间复杂度：$\mathcal{O}(D)$

3 DEG3MAXT

3.1 试题来源

codechef OCT 13 Three-Degree-Bounded Maximum Cost Subtree

3.2 题目大意

给定一张n个点，m条边的带权联通无向图，保证其中每个点双连通分量大小小于10，求张图的一个3点度限制最大生成树，即每个点点度不超过3，生成树可以不包括所有顶点，在保证边权和最大的情况下，输出方案数对2^{32}取模的结果。

3.3 算法讨论

首先，注意到对于两个点双连通分量，我们可以分开来考虑它们的3点度限制生成树，并且只需要记录下每个生成树中割点的度数就可以很容易地合并这两棵树，接下来我们考虑如何求出一个点双连通分量的3点度限制最大生成树。

由于题目限制这个分量中包含的点只有9个，那么就可以使用状态压缩动态规划来解决。记$F[s]$为所有点的度数压缩以后的结果为s时的最大生成树（此处应该要求所有度数不为0的点联通），由于每个点度数只能取$\{0,1,2,3\}$，所以s应该为一个不超过18位的二进制数，转移过程中为了避免重复计数，我们应该每次在s中选择一个度数不为0的编号最小的点来考虑，记这个点为x，若$deg_x = 1$，那么直接枚举与它相邻的点即可：

$$F[s] = \max_i \{F[s \setminus \{x:1, i:1\}] \mid deg_i > 1 \ or \ s = \{x:1, i:1\}\}$$

当$deg_x \neq 1$的时候，可以枚举与x联通的一个联通分量：

$$F[s] = \max_{a,b} \{F[a + \{x:1\}] + F[b + \{x:deg_x - 1\} \mid s = a + b + \{x:deg\}\}$$

此处应注意处理计数时重复统计的问题。

接下来，考虑多个双联通分量的情况。我们可以对于每一个联通分量单独考虑，然后对于不同的联通分量，需要记录的仅仅是它们的割点的度数。由于整个图是联通图，所以所有的点双连通分量组成了一个树形结构，这一部分用树DP来处理即可。

3.4 时空复杂度

对于每个大小为k的双联通分量，我们要枚举的是每一个合法的点度集合以及它的某一个子集。考虑到避免重复统计，所以枚举一个大小为k的集

合的子集的复杂度是$\mathcal{O}\left(3^{k-1}\right)$，对于每一个大小为$k$的点的集合，对应的点度集合应该是$\mathcal{O}\left(3^{k}\right)$个，所以总复杂度应该是$\mathcal{O}\left(9^{k-1}\right)$，考虑到度数集合的合法性，此处枚举个数为百万级别。

第二部分树DP复杂度为$\mathcal{O}(m)$

所以总时间复杂度上界为$\mathcal{O}\left(n * 9^{k-1}\right)$，其中$k$为最大的双联通分量的大小，此处$k=9$

空间复杂度：$\mathcal{O}\left(4^{n}\right)$

4 MISINT2

4.1 试题来源

codechef JAN 12 Misinterpretation 2

4.2 题目大意

求满足以下条件的长度在$[L,R]$单词的数量：将偶数位放在奇数位前面后整个单词没有变化。

$L \leq R \leq 10^{10}, Len = R - L \leq 5 * 10^4$

4.3 算法讨论

本题实际要求的内容就是对于每一个长度的单词，经过以上操作得到的环的数量。假设最终有s个环，那么答案就是26^s。接下来考虑对于一个长度如何求出经过这个置换的环的数量。

注意到对于长度为奇数的单词，经过这个变化之后最后一位是单独成环的，所以$s(2i+1) = s(2i) + 1$。考虑长度为6的置换:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 4 & 6 & 1 & 3 & 5 \end{pmatrix}$$

这个置换实际上就是$p(i) = 2i\%(n+1)$。

考虑元素i所在的环的大小，记$(i, n+1) = k, n+1 = pk$，设$(x, n+1) = k$的所有元素分别为$p_1k, p_2k \ldots p_sk$，记作集合P，那么必有$(p_i, p) = 1, p_i < p$，于是$s = \varphi(p)$。由于$(p,2) = 1, (p, p_i) = 1$，故$p_i * 2^x \in P$，且$p_i * 2^i = p_i (mod\ pk) \Leftrightarrow 2^i = 1 (mod\ p)$，故$|P| = o_2(p)$，其中$o_2(p)$表示在模$p$意义下2的阶。那么，这个环的长度就是$o_2(p)$，环中元素一共有$\varphi(p)$个。所以环的总数就是

$$\sum_{p|n+1, p\neq 1} \frac{\varphi(p)}{o_2(p)}$$

接下来考虑如何快速求出o_2，注意到$2^x = 1 (mod\ p_1), 2^y = 1 (mod\ p_2), (p_1, p_2) = 1 \Rightarrow 2^{lcm(x,y)} = 1 (mod\ p_1p_2)$。若正整数$t$满足$2^t = 1 (mod\ p_1p_2)$，那么可以推出$2^{t \bmod x} = 1 (mod\ p_1)$或$2^{t \bmod y} = 1 (mod\ p_2)$，但$x, y$为最小的满足该试的正整数，故$x|t, y|t \Rightarrow t \geq lcm(x,y)$。所以$lcm(x,y)$为最小的满足该试的正整数，即$o_2(p_1, p_2) = lcm(o_2(p_1), o_2(p_2))$。所以$o_2$为不完全积性函数。

由于$R - L \leq 5 * 10^4$，所以直接分解$R-L$个数无法通过此题。可以先预处理出不超过$\sqrt{R}$的所有质数以及质数的幂的o_2的值，然后用这些质数筛这$R-L$个数。由于$2^{\varphi(m)} = 1 (mod\ m)$，所以可以枚举$\varphi(m)$的所有约数

来求$o_2(m)$。这样求单个o_2的复杂度就是$\mathcal{O}\left(\sqrt{R}\right)$。设$\sqrt{R}$以内的质数个数为$s$，那么预处理复杂度为$\mathcal{O}\left(s*\sqrt{R}logR\right)$，筛法复杂度为$\mathcal{O}\left(s*Len*logLen\right)$。如果实现精细，那么这个复杂度就可以通过此题了。

4.4 时空复杂度

时间复杂度：$\mathcal{O}\left(s*\sqrt{R}logR+s*Len*logLen+Len\sqrt{R}\right)$

空间复杂度：$\mathcal{O}\left(\sqrt{R}logR+Len\right)$

5 YALOP

5.1 试题来源

codechef JULY 11 Trial of Doom

5.2 题目大意

给定一个$n*m$的网格，网格上有k个点是红色，其余是蓝色。你需要从$(1,1)$走到(n,m)，你可以走到一个与当前格子八联通的格子上面，每离开一个格子，这个格子以及周围的四个格子都会改变颜色。求是否存在一个方案使得到达终点时所有的格子都是蓝色。

$n \leq 40, m \leq 10^9, k \leq 10000$

5.3 算法讨论

为了讨论方便，下面称反转一个格子为将这个格子以及其周围的格子改变颜色。

考虑一个$2*2$的网格，可以发现，可以构造一种方案使得不改变任何格子的颜色移动到任意一个格子上面（只需要每个格子离开两遍，由于这四个格子两两可达，那么就可以在最后离开的那个格子直接走到自己想要到达的格子上），更进一步，可以随意反转一个格子，然后再到达任意一个格子。

那么，对于$n,m \geq 2$的情况，就可以任意反转任意格子，然后到达终点。如果所有格子的颜色以及第一列的格子是否反转确定了（这就相当于所有格子的颜色改变只能通过下一列的对应格子反转来进行），那么接下来的所有的格子的情况就可以通过上一列递推确定下来，除了最后一列，所有格子就都一定能满足颜色与最终颜色相同，所以最终是否合法就取决于最后一列的颜色情况。那么可以分别把将所有的格子的情况映射到最后一列的颜色上面，即若要这个格子的颜色改变，那么经过递推导致的最后一列的格子的颜色反转情况。这可以用一个2^n的二进制数状压下来，那么，若所有需要改变颜色的格子的映射的异或值为0，这就是一个合法的方案。由于第一列的格子的反转情况可以随意确定，所以对第一列反转进行高斯消元，再用线性基对k个格子的异或值进行判断即可。

经过观察发现这些格子映射之后的情况都存在一个不大的循环节，所以可以不用考虑m很大，只考虑循环节大小即可。

对于$min(n,m) = 1$的情况，也就是只能沿着一个方向走，如果退回，就相当于同时反转了两个格子，那么就需要额外判断最终的需要反转的格子的奇偶性。

5.4 时空复杂度

时间复杂度：$\mathcal{O}\left(n^2+ns\right)$，其中$s$为循环节大小。经过打表计算，这里不超过20000。

空间复杂度：$\mathcal{O}\left(ns\right)$

6 MAGIC

6.1 试题来源

codechef AUG 12 Two Magicians

6.2 题目大意

两个魔术师在一个有n个房间m条双向路的地方玩一个游戏。在游戏的一开始，第一个魔术师在1号房间，第二个在2号房间，每个魔术师都有p点法力。每个魔术师在自己的回合有三个步骤要做。我们假设当前回合的魔术师为A魔术师。第一步，A魔术师可以从当前房间沿着已有的路一直走，随意停下。如果他停的房间和另一个魔术师恰好相同，那么游戏结束，A魔术师胜利。第二步，A魔术师必须增加一条边原来没有的路。如果不能做到那么A魔术师失败，否则继续。第三步，如果A魔术师的法力值为正数，他可以选择消耗1点法力值，传送到任何一个房间。求先手是否必胜。

$N \leq 8000, M \leq 10000$

6.3 算法讨论

考虑最后输掉的那个魔术师的操作。最后一步一定是合并两个已经加成完全图的两个联通块使整张图联通。假设其中一个完全图的点数为x，那么从初始状态到当前状态需要加上的边数就是$n*(n-1)/2-x*(n-x)-m$。若n为奇数，那么不论x的值，边数的奇偶性就已经确定了，那么可以直接得出答案。

若n为偶数，那么就有一个人希望最后的两个联通块均为奇数，另一个人希望联通块均为偶数。当$p=0$时，例如考虑先手希望联通块均为奇数。若两人初始的联通块均为偶数，那么后手总可以保持这个状态，则先手必败。其余情况类似。当$p \neq 0$时，希望为偶数的联通块的人就多了一种选择，即将自己所在的奇数联通块以及对手所在的奇数联通块合并之后跳走，可以发现，除了n很小的特殊情况，这时期望偶数的联通块的人一定可以胜利。当一开始只有两个联通块的时候，或者一开始有三个联通块，但是自己是后手，是不能进行这个操作的，特判掉这两个特殊情况即可。

最终程序只需要dfs求出所有的联通块的大小，然后依次进行上述的判断即可。

6.4 时空复杂度

时间复杂度：$\mathcal{O}(N+M)$
空间复杂度：$\mathcal{O}(N+M)$

7 COT5

7.1 试题来源

Codechef FEB 14 Count on a Treap

7.2 题目大意

给出一个序列的值以及优先级，将整个序列构成一棵Treap，询问两点之间的路径长度或者删除、插入一个点。

7.3 算法讨论

由于优先级是给出的，所以整棵Treap可能是不平衡的。所以不能直接维护。

考虑现将序列按照值排序，那么这个序应该是Treap的中序遍历。而Treap在优先级上面满足堆性，所以这个序列中优先级的最小值就是根，然后可以递归构造出整棵Treap。

那么首先我们先求出两点的LCA，根据这个序列的性质，LCA的位置就在这两点的序列位置中间，并且优先级是最小的。

接下来，要求出一个点的深度。考虑这个点的祖先节点，根据其在序列中的位置分为两部分。对于序列中在这个点之前的点，它们一定是这个点的“左拐”的祖先，即这个点走到这个祖先的最后一条边一定是由祖先连向右儿子的。而要使一个点v成为点u“左拐”的一个祖先，就一定要满足v的优先级比$(v,u]$中的点的优先级都小。对于“右拐”祖先类似。

那么问题转化为求一个点比区间的右边的所有数都小的数的个数。考虑维护一棵线段树，线段树的每一个节点维护对应的这个区间内比右边都小的数的个数，那么考虑合并两个节点$[l,mid],[mid+1,r]$，那么$[mid+1,r]$可以直接贡献，对于$[l,mid]$，我们关心的就是$[mid+1,r]$的最小值min_r，那么定义递归过程：$calc(l,r,min_r)$，若$min[mid+1,r]<min_r$，那么min_r对于$[l,mid]$这个区间是没有影响的。则执行$calc(mid+1,r,min_r)$，否则，$[mid+1,r]$这个区间就不能贡献答案了，那么执行$calc\,(l,mid,min_r)$即可。这样每次合并会增加$logN$的询问。

对于修改，则可离线预处理。

7.4 时空复杂度

时间复杂度：$\mathcal{O}\left(Nlog^2N\right)$
空间复杂度：$\mathcal{O}\left(N\right)$

8 CLOWAY

8.1 试题来源

Codechef AUG 15 Future of draughts

8.2 题目大意

给定T张无向图，Q次询问在区间$[l_i, r_i]$的图上进行下述游戏并在K_i步之内结束游戏的方案数是多少。

这个游戏规则是这样的，首先在每张图上选定一个初始节点，接下来每一步选择任意多个图（不能为0个），然后对于选中的图，将这些图的节点移动到一个相邻节点上，当所有节点回到初始位置时游戏结束。两个方案不同即初始节点不同或存在一步移动不同。

8.3 算法讨论

首先考虑只有一张图的时候怎样求出答案。

那么问题就是求出一张图中有多少条闭合回路长度不超过K。也就是对于$1 \leq i \leq K$求出有多少条闭合回路长度为i。也就是对于一张图的邻接矩阵G，求出G^i的对角线元素之和。

我们称对角线元素之和为矩阵的迹，记作$tr[G^i]$。由线性代数的知识可知：$tr[G] = \sum_i \lambda_i$，其中λ_i为矩阵的N个特征值。这个关系可由矩阵的行列式中λ^{N-2}的系数得到。那么由于λ^i也是G^i的特征值，那么$tr[G^t] = \sum_i {\lambda_i}^t$。于是我们可以先求出$tr[G^t] = \sum_i {\lambda_i}^t, 0 \leq t \leq N$（直接矩阵乘法），然后用牛顿等幂和公式在$\mathcal{O}(N * K)$的时间内求出$tr[G^t] = \sum_i {\lambda_i}^t$，在$N + 1 \leq t \leq K$的答案。

对于多张图的情况，strong products of graphs的知识，可以得出以下关系：

$$tr[(F * G)^k] = \sum_{i=0}^{k} \binom{k}{i} * (-1)^{k-i} * tr[(F + I)^i] * tr[(G + I)^i]$$

这个结论也可以推广到多张图的情况。

注意到以上的式子是可以用卷积表示的，故可以用快速傅里叶变换优化。

8.4 时空复杂度

时间复杂度：$\mathcal{O}\left(T^2 * K * logK + T * N^4 + TNK\right)$

空间复杂度：$\mathcal{O}(TNK)$

9 SPMATRIX

9.1 试题来源

codechef JUNE 13 Count Special Matrices

9.2 题目大意

$N(N \geq 3)$是一个正整数。A是一个$N * N$的整数矩阵，x行第y个元素记作$A[x][y]$，这个矩阵如果满足以下条件就称它是特殊的:

- $A[x][x] = 0$ for $1 \leq x \leq N$
- $A[x][y] = A[y][x] > 0$ for $1 \leq x < y \leq N$
- $A[x][y] \leq max(A[x][z], A[z][y])$ for $1 \leq x, y, z \leq N$
- $A[x][y] \in \{1, 2, ..., N-2\}$ for $1 \leq x < y \leq N$
- $\forall k \in \{1, 2, ..., N-2\}, \exists x, y \in \{1, 2, ..., N\}, A[x][y] = k$

给出Q组询问，询问大小为N的特殊矩阵个数。

9.3 算法讨论

由1、2两点，这个矩阵可以认为是一个无向图矩阵，第三点可以认为这个矩阵保存的是图上两点间路径的最大值。那么求这种矩阵的个数，也就是求满足条件的图的点对最大值的情况的个数。这里需要注意的是，即使图不想同，但是有可能最后表示图两点之间最大值的矩阵相同，这个时候只能统计一次。

考虑按照边权从小到大加入边，由于每种边权至少出现一次，且一共有$N-2$个边权，那么每中边权就最多出现两次。用$F[i][0]$表示i个点，$i-1$种边权的方案数，$F[i][1]$表示，i个点，$i-2$种边权的方案数，那么转移就可以考虑当前这种边权出现了多少次，然后选择几个联通块连接起来即可。所以最后得到:

$$F[i][0] = F[i-1][0] * \binom{n-1}{2}$$
$$F[i][1] = \left(3 * \binom{n}{4} + \binom{n}{3}\right) * F[i-1][0] + \binom{n}{2} * F[i][1]$$

由于这一题的N的范围很大，所以在递推过程中应该尽量避免乘法，即使用公式$\binom{n}{i} = \binom{n-1}{i} + \binom{n-1}{i-1}$来递推计算组合数。

9.4 时空复杂度

时间复杂度：$\mathcal{O}(N + QlogQ)$
空间复杂度：$\mathcal{O}(N)$

10 CHEFBOOK

10.1 试题来源

Codechef JUNE 15 Chefbook

10.2 题目大意

给定一张有向图，每个条边有边权，你可以进行两种操作：1.把一个点的所有的出边权值加上任意整数。2.把一个点入边边权减去任意整数。给定每条边的最终边权的范围$[L_i, R_i]$，要求所有操作结束后每条边的边权都落在这个范围内。现在要求你求出最大边权和，并输出方案。

10.3 算法讨论

注意到这些限制都可以写成线性规划的形式：每条边的边权记为v_i，每一个点拆为两个操作的变量P_i, Q_i,那么限制就是

$$L_i \leq v_i + P_{from} - Q_{to} \leq R_i$$

，要求最大化

$$\sum_i outdeg_i * P_i - indeg_i * Q_i$$

直接用单纯形法无法通过此题全部数据。所以我们考虑这个问题的对偶问题。将整个线性规划矩阵转置，由于原先的限制中每一行只有两个变量，那么转置矩阵的每一列也就只有两个位置为不为0，分别为$+1, -1$，这与网络流问题的限制矩阵相似，因为网络流问题要求每一个点的入度等于出度，而一个单位流量恰好贡献了一个入度以及一个出度。那么原问题的最大值就对应着对偶问题的最小值。

经过仔细分析，不难得出网络流问题的建图。然后可以用费用流解决这个问题。接下来考虑如何输出方案。

由于网络流的边表示的是原问题的一些限制条件，那么有流量的边就对应到原问题的限制取到最值（也就是取到等号），没有流量的边就表示原限制没有取到等号。所以可行方案就转化为在满足一些型为$x_i - x_j \leq t$的基础上的一组可行解。注意到这与最短路问题的距离顶标的限制相同，t即为一条连接i, j两点的有向边。那么直接使用最短路算法即可解决。

10.4 时空复杂度

为输入同阶的费用流复杂度。

11 EST

11.1 试题来源

codechef JULY 12 Equivalent Suffix Tries

11.2 题目大意

给出一个串S，长度为N，求有多少个串与这个串构出的后缀树相同。（此处后缀树没有加入特殊字符作为结尾）。

$N \leq 10^5$

11.3 算法讨论

设L为最大的一个自然数，满足$S[N-L+1,N]$为S的某一个后缀suf_j的前缀，其中满足$j \leq N-L$。那么考虑暴力构造后缀树时，从长到短插入后缀，前$N-L$个后缀都可以贡献一个叶子节点（否则可以找到更大的L），后L个后缀由于相等关系，不能贡献叶子节点。故后缀树的叶子节点等于$N-L$是一个定值。

由于前$N-L$个后缀是叶子，而在后缀树上深度确定的叶子节点是唯一的，所以这$N-L$个字母的相等关系必然也被保留，且这$N-L$个后缀两两之间的lcp也必然是定值（因为后缀树的形态确定，故两两叶子的lca也应确定）。记$Q = max_{1 \leq i < N-L} \{lcp(i, N-L)\}$，那么前$N-L+Q-1$个字母的相等关系就已经确定了。且第$N-L+Q$个字母将会有一些不等关系来保证$Q$是最大的自然数。

于是，只需要保证 1.前$N-L+Q-1$个字母的相等关系；2.第$N-L+Q$个字母的不等关系；3.$S[N-L+1,N]$为某一后缀的前缀 即可保证后缀树同构。

那么，可以枚举满足条件的suf_j，但是不同的j有可能得到相同的串，所以要用hash判断两个串是否相同。考虑填充字母，当$j+L-1 < N-L+1$时，可以直接计算hash。否则，填充的串一定是循环的，循环节为$N-L-j$。这里可以直接暴力求出hash，每一次求出hash值复杂度为$\mathcal{O}(L/(N-L-j))$，那么总复杂度就为$\mathcal{O}(LlogL)$

11.4 时空复杂度

时间复杂度：$\mathcal{O}(NlogN)$

空间复杂度：$\mathcal{O}(N)$

12 SHORTCIR

12.1 试题来源

codechef AUG 11 Shortest Circuit Evaluation

12.2 题目大意

给出一个含有n个变量，三个运算（and,or,not）的布尔表达式，其中每个变量都给出为真的概率，求一个计算顺序，使得计算这个布尔表达式需要访问的变量的个数的期望值最小。

$n \leq 1000$

12.3 算法讨论

首先可以暴力构造出这个布尔表达式的后缀表达式树，由于相同的运算满足交换律，所有这里需要将连续的一段相同的运算合并到同一个节点上面。然后对于每一个节点，只需要确定其儿子的计算顺序即可。

由于not节点只有一个儿子，所以可以直接计算。

不妨考虑一个and节点。假设已经计算出了它的所有儿子为真的概率p_i，以及计算这个儿子的子树的期望E_i。那么这个节点为真的概率恒等于$\prod_{i=1}^{s} p_i$需要将它的所有儿子排序，使得

$$E = \sum_{i=1}^{s} \left(\prod_{j=1}^{i-1} p_j \right) * E_i$$

最小。考虑调整法交换两个儿子的顺序，假设这两个儿子i, j相邻，那么这两个儿子交换之后对于i之前的计算，以及j之后的计算都不会产生影响，那么i在j之前计算的条件就是

$$\begin{aligned} E_i + p_i * E_j &\leq E_j + p_j * E_i \\ \frac{1}{E_j} + \frac{p_i}{E_i} &\leq \frac{1}{E_i} + \frac{p_j}{E_j} \\ \frac{1}{E_j} - \frac{p_j}{E_j} &\leq \frac{1}{E_i} - \frac{p_i}{E_i} \end{aligned}$$

故只需要按照$\frac{1}{E_j} - \frac{p_j}{E_j}$排序，然后依次计算即可。

对于or节点，可以将其转化为and节点，即

$$b_1 \ or \ b_2 \ or \ \ldots or \ b_n = not((not \ b_1) \ and \ (not \ b_2) \ and \ \ldots \ and \ (not \ b_n))$$

12.4 时空复杂度

时间复杂度：$\mathcal{O}(nlogn)$
空间复杂度：$\mathcal{O}(n)$

13 REALSET

13.1 试题来源

Codechef DEC 13 Petya and Sequence

13.2 题目大意

给出一个长度为N的序列A_i，问是否存在一个长度为N的序列B_i，满足：

- 至少存在一个$0 \le i \le N-1$满足$B_i \neq 0$
- 对于任意$0 \le j \le N-1$满足 $\sum_{i=0}^{N-1} A_i * B_{(i+j) \bmod n} = 0$

13.3 算法讨论

记

$$|A| = \begin{vmatrix} a_0 & a_1 & \cdots & a_{N-1} \\ a_{N-1} & a_0 & \cdots & a_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & \cdots & a_0 \end{vmatrix}$$

则原问题即求$|A| = 0$是否成立。

构造

$$|B| = \begin{vmatrix} 1 & 1 & \cdots & 1 \\ \omega_0 & \omega_1 & \cdots & \omega_{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ {\omega_0}^{N-1} & {\omega_1}^{N-1} & \cdots & {\omega_{N-1}}^{N-1} \end{vmatrix}$$

其中ω_i为1的i次单位根，满足${\omega_i}^i = 1$。记$f(x) = \sum_{i=0}^{N-1} a_i * x^i$，则

$$|AB| = \begin{vmatrix} f(\omega_0) & f(\omega_1) & \cdots & f(\omega_{N-1}) \\ \omega_0 f(\omega_0) & \omega_1 f(\omega_1) & \cdots & \omega_{N-1} f(\omega_{N-1}) \\ \vdots & \vdots & \ddots & \vdots \\ {\omega_0}^{N-1} f(\omega_0) & {\omega_1}^{N-1} f(\omega_1) & \cdots & {\omega_{N-1}}^{N-1} f(\omega_{N-1}) \end{vmatrix} = \prod_{i=0}^{N-1} f(\omega_i) * |B|$$

所以，

$$|A| = \prod_{i=0}^{N-1} f(\omega_i) = \prod_{i=0}^{N-1} \sum_{j=0}^{N-1} a_j * {\omega_N}^{ij}$$

构造

$$F(x)=\sum_{i=0}^{N-1}a_i*{\omega_N}^{i*i}*x^i, G(x)=\sum_{i=-N+1}^{N-1}{\omega_N}^{-i*i}*x^i$$

则

$$(F(x)*G(x))[i]=\sum_{j=0}^{N-1}a_j*{\omega_N}^{2*i*j-i^2}$$

故可以用FFT加速求出$|A|$。

为了避免精度问题，可以选择在模意义下进行计算。

13.4 时空复杂度

时间复杂度：$\mathcal{O}(NlogN)$

空间复杂度：$\mathcal{O}(N)$

14 MAXCIR

14.1 试题来源

codechef OCT 12 Max Circumference

14.2 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x_i, y_i，使用这个操作可以使得点A的x坐标增加x_i，并且y坐标增加y_i。你可以使用最多K个操作，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。最大化三角形ABC的周长，答案的绝对误差必须小于10^{-12}。

$K \leq N \leq 500, |x_i|, |y_i| \leq 10^6, |x|, |y| \leq 10^9$

14.3 算法讨论

首先，$|BC|$是定值可以被忽略。那么考虑最大化$|AC| + |BC|$，若得到的最大值为ans，那么对于所有解都满足$|AC| + |BC| \leq ans$，也就是A点全部在一个椭圆内部。记A'点为最优点，那么A'就在椭圆的边界上，考虑在A'点的切线$f(x, y) = c$，那么对于所有在椭圆内部的点，都满足$f(x, y) \leq c$，于是，只需要最大化$f(x, y)$即可找到这个A点。也就是对于任意一条向量v，然后选取$op_i * v$最大的K个向量。容易发现，对于$(x_1, y_1), (x_2, y_2)$，它们点乘一条向量的大小关系变化的分界点是$(y_2 - y_1, x_1 - x_2)$。

考虑处理出所有的分界点，然后对这些临界向量极角排序，每次大小关系变化时更新以下选择的操作即可得到答案。

此题精度要求极高，可以将一个小数分为I, D两个部分，分别为整数部分以及小数部分，开根号时可以用下述方法:

$$S = (I + D)^2 = I^2 + 2ID + D^2$$
$$D = \frac{S - I^2}{2I + D} = \frac{S - I^2}{I + S}$$

，这里的S用系统函数开根号即可保证精度。

14.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N^2 logN\right)$
空间复杂度：$\mathcal{O}\left(N^2\right)$

15 GERALD08

15.1 试题来源

Codechef APRIL 14 Chef and Tree Game

15.2 题目大意

给出一棵树，树边为黑或白。现在A,B两个人在树上博弈，每次每个人删去一条边，并将不与根相连的块删去。不能操作者为负。要求A只能删黑边，B只能删白边。输出AB分别先手时的获胜情况。

15.3 算法讨论

这是经典的Hackenbush问题。对于一个局面S，定义一个与之对应的surreal number $S = \{S_L|S_R\}$，S_L表示A执行操作以后的所有局面的集合，S_R类似。一个surreal number可以对应一个实数，两个游戏的和即为实数相加。若最终整棵树的局面对应的实数$x = 0$，则先手败，若$x < 0$则A败，否则B败。

根据Hackenbush的结论，一个局面$G, x = \{x_L|x_R\}$，那么将整个树G置于一条边上（即加入一条边与原树根相连，并且新树根为改边连接的另一个节点），若这条边为黑色，那么得到的局面G'对应的surreal number就应该是$\frac{x+p}{2^p-1}$，其中p为使得$x + p > 1$的最小正整数。若这条边为白色，那么得到的局面应该是$\frac{x-p}{2^p-1}$，其中p为使得$x - p < -1$的最小正整数。

由于精度问题，显然不能用实数运算。由于surreal numbers对应的实数一定是某个二进制小数，那么维护一个链表表示该小数，进行加法操作的复杂度就是$min\{p, q\}$，其中p, q为两个加数的位数。又，每个点i最多使整个数增加deg_i位，所以类似与启发式合并，总复杂度应该为$\mathcal{O}(N * logN)$

15.4 时空复杂度

时间复杂度：$\mathcal{O}(NlogN)$
空间复杂度：$\mathcal{O}(N)$

16 TRIPS

16.1 试题来源

Codechef OCT 14 Children Trips

16.2 题目大意

给定一棵N个节点的树，每一条边的边权为1或2，给定起点、终点以及每天能行走的最大距离，要求每一天不能停在路上，求最少要多少天能到达。询问有M组。

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

16.3 算法讨论

容易发现，若一个人可以在k天的时间内从u到v，那么这个人肯定可以在k天从v到u。所以，就可以计算从起点、终点一起出发走到lca的天数。同时我们发现，若一个人的体力值很小，那么我们可以预处理答案，若一个人的体力值很大，那么显然在不多的步数内就可以直接到达，所以这启发我们分开处理两类询问。

首先我们考虑一个人的体力值$x \geq t$的情况。

这一部分由于答案不大，那么我们可以直接模拟一步一步地跳。使用倍增的方法，暴力处理答案即可。应该注意的是从两边往上跳的时候，最后一步可能没有跳满，应该两边合起来考虑。 这一部分复杂度为$\mathcal{O}(N/t * logN)$

接下来考虑$x \leq t$的情况。

由于x并不大，所以对于每一个x我们可以预处理出整个倍增数组，这个倍增数组$f[i][j]$表示的是i号点向上跳2^j天能到达的点。那么对于每一个询问，直接倍增求出答案即可。这一部分复杂度为$\mathcal{O}(t * N * logN)$

16.4 时空复杂度

时间复杂度：当$t = \sqrt{N}$时，复杂度最小，为$\mathcal{O}\left(N * \sqrt{N} * logN\right)$

空间复杂度：$\mathcal{O}\left(N * \sqrt{N}\right)$

17 GRAPHCNT

17.1 试题来源

Codechef MAY 15 Counting on a directed graph

17.2 题目大意

给出一张有向图，询问有多少对点(x,y)满足存在两条点不相交的路径分别从$1 \to x, 1 \to y$。

17.3 算法讨论

首先，定义一个点x的$idom(x)$为，$1 \to x$的每条路径都经过$idom(x)$，且$idom(x)$是在所有满足第一个条件下DFS标号最大的点。

可以证明，$idom$形成了一个树形结构，并且当且仅当$idom(x), idom(y)$在这棵树上的最近公共祖先为根，那么存在两条点不相交的路径分别从$1 \to x, 1 \to y$。

为了求出$idom(x)$，再定义$sdom(x)$为，存在一条从$sdom(x) \to x$的路径，满足路径上除了起点和终点的所有节点的DFS编号均大于x，那么由定义可知，$idom(x)$一定为$sdom(x)$在DFS树上的祖先。类似强联通分量，我们可以发现若$sdom(x) \to x$路径上的点u，全部满足$sdom(u) > sdom(x)$，则$idom(x) = sdom(x)$，否则，记$sdom(u)$最小的节点为v，那么$idom(x) = idom(v)$。

为了求出$sdom(u)$，可以将点按照DFS的逆序插入，这样已经插入的点的编号就都大于未插入的点的编号了。若存在一条边$v \to u$，那么v到其已经插入的所有祖先的$sdom$都可以贡献给u。维护DFS树的路径信息，可以用带权并查集。

17.4 时空复杂度

时间复杂度：$\mathcal{O}(NlogN)$[2]
空间复杂度：$\mathcal{O}(N)$

[2]此处的并查集维护路径信息，不能使用按秩合并，否则会破坏祖先关系，但是tarjan在论文中给出了另一种复杂度为$\mathcal{O}(N\alpha)$的实现

18 GTHRONES

18.1 试题来源

codechef AUG 12 A Game of Thrones

18.2 题目大意

给出N个不同的数，第i个数为u_i，出现次数为c_i。两个人进行博弈，先手选择一个数u，之后每一轮，玩家选择一个数v，要求$u/v, v/u$中至少有一个数为质数。然后擦去u，当前的数变为v。求先手是否必胜，必胜的情况下，能选择哪些数作为第一个数。

$N \leq 500, u_i \leq 10^{18}, c_i \leq 10^9$

18.3 算法讨论

首先，将两个数中可以转移的连边构成一张图，显然这个图是二分图（因为相邻两个点只相差一个素因子）。先考虑$c_i = 1$的情况。二分图博弈有如下结论：先手必败当且仅当这张图存在完备比配。因为后手始终可以沿着匹配边进行游戏。先手不能跳出这个匹配。

当不存在完备匹配时，考虑二分图上的一个最大匹配。若u点可以不在最大匹配上，那么先手就可以选择u。这时后手选择的v必定在最大匹配上（否则就找到了一条增广路增加最大匹配），那么先手就可以沿着匹配边一直走下去。

$c_i > 1$时，只需要用网络流进行最大匹配，并把每个点的权变为出现次数即可。

为了判断一个点是否可以不在最大匹配上，首先若一个点本身就没有满流，那么它显然可以不在最大匹配上。否则，就需要从这个点退流。那么将残量网络拿出来bfs一遍就能确定每个点是否可以退流了。

由于$u \leq 10^{18}$，所以判断素数的时候要使用Miller-Rabin算法。

18.4 时空复杂度

时间复杂度：$maxflow(N, N^2)$

空间复杂度：$\mathcal{O}\left(N^2\right)$

19 LUCKYDAY

19.1 试题来源

codechef NOV 11 Luckdays

19.2 题目大意

给出数列$s_1 = a, s_2 = b, s_i = x * s_{i-1} + y * s_{i-2} + z \mod p$，求$i \in [l, r]$中，$s_i = c$的$i$的个数。

$p \leq 10007, l \leq r \leq 10^{18}$

19.3 算法讨论

若$x = 0, y = 0$，那么这个数列就是常数数列，直接计算即可。

若$y = 0$，那么这个数列的循环节不会超过p，暴力计算出循环节即可。

否则，数列的循环节不超过p^2。

那么考虑数列的递推矩阵：

$$\begin{pmatrix} x & y & z \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} s_i \\ s_{i-1} \\ 1 \end{pmatrix} = \begin{pmatrix} s_{i+1} \\ s_i \\ 1 \end{pmatrix}$$

记为$A * u_i = u_{i+1}$，为了求出循环节，只需要求出$A^t * u_0 = u_0$的最小正整数t即可。由于矩阵可逆，所以这里可以用大步小步法求出，即设$t = k * m - b$，则$A^{km} * u_0 = A^b * u_0$，预处理出$A^b * u_0$的所有情况插入hash表中，然后枚举k即可。复杂度$\mathcal{O}\left(m + p^2/m\right)$。

接下来考虑如何求出C的出现次数，枚举x，使得$u_i = (C, x, 1)^T$，那么也使用上述方法即可求出u_i的出现位置，对于$i \leq R$的时候，可以将$[1, R]$分为一段循环节加上一段不完整的循环，不完整的部分可以二分求出位置，那么求得$i \in [1, R]$的个数就很方便了。这一部分复杂度$\mathcal{O}\left(p * p^2/m\right)$

取$m = p^{1.5}$即可得到总复杂度$\mathcal{O}\left(p^{1.5}\right)$

19.4 时空复杂度

时间复杂度：$\mathcal{O}\left(p^{1.5}\right)$

空间复杂度：$\mathcal{O}\left(p^{1.5}\right)$

20 KNGHTMOV

20.1 试题来源

codechef SEP 12 Knight Moving

20.2 题目大意

考虑一张无限大的方格棋盘。我们有一个“骑士”，从$(0,0)$格开始，移动至(X,Y)格：它只能从(u,v)格移动至$(u+Ax,v+Ay)$或者$(u+Bx,v+By)$。棋盘上有K个障碍格，骑士不能进入这些格子。 计算骑士有多少种到达指定位置的方案。

$K \leq 15, |X|, |Y|, |Ax|, |Ay|, |Bx|, |By| \leq 500$

20.3 算法讨论

首先考虑向量$(Ax,Ay),(Bx,By)$线性无关时的计算。由于移动向量线性无关，所以棋盘的所有位置(x,y)均可以表示为唯一的点对(u,v)，使得$(x,y)=u(Ax,Ay)+v(Bx,By)$，所以，问题转化为在一个方形网格中，可以向右或向上移动一格，有K个障碍，求到达右上角的方案数。注意这时坐标范围有可能达到$|X|^2$级别，所以不能直接递推计算。

考虑使用容斥原理，首先将障碍格排序，记$f[i]$为从坐下到第i个障碍，除了终点不经过任何一个障碍的方案数。那么可以对一条不合法路径经过的第一个障碍进行分类统计，最终得到，

$$f[i] = ways(x_i, y_i) - \sum_{j=1}^{i-1} ways(x_i - x_j, y_i - y_j) * f[j]$$

其中$ways(x,y)$表示从$(0,0)$到(x,y)的方案数，即$\binom{x+y}{x}$。

接下来，若向量$(Ax,Ay),(Bx,By)$线性相关，那么能到达的点就是一条直线上的点。若能在这条线上面走出一个环，那么答案就是∞，判环可以dfs暴力。若不存在环，那么转移图就是一个DAG，直接DP计数即可。

20.4 时空复杂度

时间复杂度：$\mathcal{O}\left(|X|^2 + K^2\right)$
空间复杂度：$\mathcal{O}\left(|X|^2 + K\right)$

21 BB

21.1 试题来源

codechef JULY 11 Billboards

21.2 题目大意

一个长度为n的序列，你需要给某些位置填上1，要求任意连续m个位置都有不小于k个1，并且1的总数最小，求方案数。

$1 \leq k \leq m \leq 500, m \leq n \leq 10^9$

21.3 算法讨论

首先考虑$m \mid n$的情况。这时将序列分为$\frac{n}{m}$段，对于每一段必然只有k个1（否则1的个数不是最少的），那么用一个$\frac{n}{m} * k$的矩阵来表示这些1的位置。例如对于$n = 9, m = 3, k = 2$，摆放情况为011101110的一个序列，可以将其表示为

$$\begin{pmatrix} 2 & 3 \\ 1 & 3 \\ 1 & 2 \end{pmatrix}$$

其中每一行表示这一块中的1在这一块中的相对位置。那么容易发现这个矩阵一定是从左到右严格递增，从上到下非严格递降。并且，任意一个满足以上条件的矩阵就一定表示了一个合法的摆放情况。也就是只需要对矩阵计数即可。矩阵方案数公式如下：

$$\prod_{(i,j)} \frac{r-j+i}{\frac{n}{m}+k-i-j+1}$$

其中m表示候选集合大小，这个公式可以通过群的知识计算得到。可以发现上述式子在j不变的时候，i的有效项数只有m项，其余的部分可以直接约分，所以暴力计算即可。

考虑$m \nmid n$的情况，若$n\%m \leq m - k$，那么每一组的前$n\%m$个位置一定都是0，否则每一组的后$m - n\%m$个位置一定都是1，所以就可以转化为$m \mid n$的情况计算。

21.4 时空复杂度

时间复杂度：$\mathcal{O}(km)$

空间复杂度：$\mathcal{O}(1)$

22 SHORT

22.1 试题来源

codechef SEPT 11 Short

22.2 题目大意

给你两个数n,k，你需要找出所有的数对(a,b)，满足$n<a<k,n<b<k$，并且$(a-n)(b-n)\mid ab-n$

$n\leq 10^5,k\leq 10^{18}$

22.3 算法讨论

首先题目等价于求$0<a<k-n,0<b<k-n,ab\mid n(a+b+n-1)$的$(a,b)$的对数。

当$n=0$时，答案就是$(k-n-1)^2$，接下来考虑$n\neq 0$的情况。

不妨设$a\leq b$，$ab\mid n(a+b+n-1)\Rightarrow kab=n(a+b+n-1)$。由于$ab\geq a^2$，所以等式左边是平方增长的，等是右边是线性增长的，故a不会很大，经过简单计算，$a\leq 2.42*n$。

那么，不妨枚举a。接下来就是求b的个数，由等式可以得到

$$b=\frac{n(a+n-1)}{ka-n}$$

当a不是很大的时候，可以分解$n(a+n-1)$来求出每一个可能的b。$n(a+n-1)$的约数个数为$\mathcal{O}(n)$个，设定一个阈值t，当$a\leq t$的时候通过分解因数来求b，这一部分复杂度为$\mathcal{O}(t*n)$。

当$a>t$的时候

$$kab\leq n(a+b+n-1),a\leq b\Rightarrow k\leq 2*\frac{n}{a}+\frac{n^2}{a^2}$$

，那么可以暴力枚举k来求出所有可能的b。这一部分复杂度为

$$\mathcal{O}\left((2.24*n-t)*\frac{n^2}{t^2}\right)=\mathcal{O}\left(\frac{n^3}{t^2}\right)$$

故当t取到$\mathcal{O}\left(n^{\frac{2}{3}}\right)$时，复杂度达到最优。

22.4 时空复杂度

时间复杂度：$\mathcal{O}\left(n^{\frac{5}{3}}\right)$

空间复杂度：$\mathcal{O}(n)$，这里是由于分解需要预处理所有质数。

23 DIVISOR

23.1 试题来源

codechef AUG 11 Something About Divisors

23.2 题目大意

对于给定的正整数B和X，求满足条件的正整数N的个数:要求对于N，至少存在一个数$D(N < D \leq B)$能整除$N * X$。共T组询问。

$T \leq 40, B \leq 10^{12}, X \leq 60$

23.3 算法讨论

记$K = \frac{NX}{D}$，那么$D < N \leq B, D \mid NX \Rightarrow K < X, N \leq \frac{BK}{X}, K \mid NX$。这样转换之后，先枚举$K$，那么$K \mid NX \Rightarrow \frac{K}{gcd(K,X)} \mid N$，记$C_K = gcd(K, X), R = \frac{BK}{X}$，那么满足条件的$N$的个数就是$\frac{R}{C_K}$。

但是同一个N有可能会被多个K计算。所以需要扣除重复计算的部分，根据容斥原理:

$$ans = \sum_{K=\{K_1,K_2 \ldots K_s\}} (-1)^s * \frac{R}{lcm\left(C_{K_1}, C_{K_2} \ldots C_{K_s}\right)}$$

直接根据这个式子计算复杂度为$\mathcal{O}\left(X * 2^X\right)$，显然无法接受。但是注意到$C_{K_i} \leq X$，所以可能出现的$lcm$的值不会很多，对于每个质数$p$，可能出现的最大幂次就是$\lfloor log_p X \rfloor$，那么，总共有可能出现的$lcm$的方案数$state = \prod_{i \in P, i \leq X} \lfloor log_p X \rfloor + 1$，在$X \leq 60$的时候$state \leq 2 * 10^5$。那么可以用一个bfs过程来计算并且统计所有可能的$state$的值以及在答案中的系数。

使用上述算法需要一定的常数优化才能通过此题。

23.4 时空复杂度

时间复杂度：$\mathcal{O}\left(state * X * T\right)$

空间复杂度：$\mathcal{O}\left(state * X\right)$

24 FN

24.1 试题来源

codechef OCT 13 Fibonacci Number

24.2 题目大意

输入两个数C, P，求最小的n，使得$F_n \equiv C\,(modP)$，或指出无解，其中F_n为斐波那契数列的第n项。

$0 \leq C \leq P-1,\ 11 \leq P \leq 2*10^9$，其中$P$以及$P\ mod\ 10$为质数。

24.3 算法讨论

首先，由于$P\ mod\ 10$为质数，我们可以推断$P \equiv \pm 1\,(mod\ 5)$，那么根据二次剩余的知识，在模P意义下一定存在5的平方剩余。

那么，由斐波那契数列通项公式可得:

$$F_n = \frac{1}{\sqrt{5}} * \left(\left(\frac{\sqrt{5}+1}{2} \right)^n - \left(\frac{1-\sqrt{5}}{2} \right)^n \right)$$

记

$$\phi = \frac{\sqrt{5}+1}{2}, \phi^{-1} = \frac{\sqrt{5}-1}{2}$$

那么

$$F_n = \frac{1}{\sqrt{5}} * \left(\phi^n - (-\phi)^{-n} \right)$$

所以

$$\frac{1}{\sqrt{5}} * \left(\phi^n - (-\phi)^{-n} \right) = C$$

当n为偶数时

$$\phi^n - \phi^{-n} = C * \sqrt{5}$$

$$\phi^{2*n} - C * \sqrt{5} * \phi^n - 1 = 0$$

由二次方程求根公式，可得:

$$\phi^n = \frac{C \pm \sqrt{C^2+4}}{2}$$

所以，在此可以用二次剩余的知识直接求出ϕ^n，接着用大步小步法求出n，判断是否符合假设即可。n为奇数时类似。

24.4 时空复杂度

时间复杂度：$\mathcal{O}\left(\sqrt{P}\right)$

空间复杂度：$\mathcal{O}\left(\sqrt{P}\right)$

25 COOLNUM

25.1 试题来源

codechef JUNE 12 Cool Numbers

25.2 题目大意

一个数有k位，那么记每个数位上的数字分别为$x_1, x_2, ..., x_k$，如果一个数n存在一到三个数位上的数的和为s，且$(x_1+x_2+...+x_k-s)^s$是n的倍数，那么n是cool number。定义$LC(n)$和$RC(n)$分别是小于等于n的最大的cool number和大于n的最小的cool number，多次询问给定n，求$LC(N)$和$RC(N)$。

$N \leq 10^{1000}$

25.3 算法讨论

注意到$s \leq 27$，那么满足$(x_1 + x_2 + ... + x_k - s)^s > n$的最大的$n$为$10^{80}$左右。那么$sum = \sum_{i=1}^{k} x_i \leq 720$。于是，可以直接枚举$sum$，然后暴力求出$sum^{27}$的所有因数，判断它们是否是cool number。由于720以内的数最多含有4个质因子，那么枚举的因数最多就只有27^4个，所以这一部分的复杂度是可以承受的。

求出所有的cool number后发现只有不到40000个。那么可以对这些cool number排序，然后每次询问二分查找即可。

另一类的cool number是满足只有不超过三个数位上的数非零，那么就能满足$sum - s = 0$从而成为cool number。这一部分可以直接从输入的n计算出来，复杂度为$\mathcal{O}(len)$

25.4 时空复杂度

时间复杂度：$\mathcal{O}(T(slogs + len) + C)$，其中$s$为第一部分的cool number的数量，C为预处理复杂度，即搜索复杂度，为玄学。

空间复杂度：$\mathcal{O}(s * len)$

26 EASYEX

26.1 试题来源

Codechef JULY 15 Easy exam

26.2 题目大意

现在有一个可以掷出$[1..K]$的色子。你扔这个色子N次，记a_i为扔到i的次数。求$\prod_{i=1}^{L} {a_i}^F$的期望（在对2003取模的情况下）。

26.3 算法讨论

首先我们考虑$F = 1$的情况。考虑下面的式子：

$$E[(x_{1,1} + x_{1,2} + ...x_{1,N}) * (x_{2,1} + x_{2,2} + ...x_{2,N}) * ... * (x_{L,1} + x_{L,2} + ...x_{L,N})]$$

这个式子中，$x_{1,2}$表示的是在第而次扔到数字1的事件。那么，我们可以将这个式子展开，注意到$x_{i,k} * x_{j,k}$在$i \neq j$时对答案的贡献应该是0，因为这两个事件不能同时发生。

经过简单的数学推导，这一部分的答案应该是$\frac{N!}{(N-L)!*K^N}$。

若$F > 1$，那么我们的式子就是：

$$E[(x_{1,1}+x_{1,2}+...x_{1,N})^F*(x_{2,1}+x_{2,2}+...x_{2,N})^F*...*(x_{L,1}+x_{L,2}+...x_{L,N})^F]$$

将这个式子展开以后，可以将$x_{1,i}$的部分统一计算方案数，设其中含有不同的i有j个，那么将其记为$num_way[F, i]$，那么我们就是要计算下述多项式中，x^F的系数

$$(num\_ways[F, 0]x^0 + num\_ways[F, 1]x^1 + ... + num\_ways[F, F]x^F)^L$$

这里可以用FFT优化多项式乘法。其中$num_ways[i, j]$可以用DP在F^2的时间求出。

26.4 时空复杂度

时间复杂度：$\mathcal{O}\left(L * F * logF + F^2\right)$
空间复杂度：$\mathcal{O}\left(L * F + F^2\right)$

27 DISTNUM

27.1 试题来源

Codechef AUG 15 Simple Queries

27.2 题目大意

给一个长度为N的数组，以及Q个操作，支持插入、删除、修改一些位置上的数。询问$[l,r]$中

$$\sum_{1\le i<j<k\le |S|} S_i * S_j * S_k$$

的值，S为区间$[l,r]$中所有数的不可重集合，或者询问$|S|$。

27.3 算法讨论

首先，考虑$|S|$如何维护，可以构造一个数组pos_i，表示与第i个位置上出现的数相同的数的上一次出现位置。那么把$i \in [l,r]$中$pos_i < l$的位置数出现的所有数的次数，因为一种数只会统计一次。这可以用二位数据结构维护。

那么，自然可以想到对于维护$\sum_{1\le i<j<k<|S|} S_i * S_j * S_k$也可以用同样的方法。记:

$$Sum_1 = \sum_{1\le i\le |S|} S_i, Sum_2 = \sum_{1\le i\le |S|} {S_i}^2, Sum_3 = \sum_{1\le i\le |S|} {S_i}^3$$

那么

$$\sum_{1\le i<j<k\le |S|} S_i * S_j * S_k = \frac{1}{6} * \left({Sum_1}^3 - 3 * Sum_1 * Sum_2 + 2 * Sum_3\right)$$

于是，对于每一个树上节点只需要维护这个点的范围内的权值的1~3次方和即可。快速找到一种数字前一个出现的位置可以用set直接维护。

由于数据范围较大，而且一次操作最多需要改动三个位置，所以要应用不少常数优化技巧。

27.4 时空复杂度

时间复杂度：$\mathcal{O}\left(n * log^2 n\right)$
空间复杂度：$\mathcal{O}\left(n * log^2 n\right)$

28 CUCUMBER

28.1 试题来源

codechef JAN 13 Cucumber Boy and Cucumber Girl

28.2 题目大意

在模2意义下给出B个$N*N$的矩阵F_i，求满足以下条件的(i,j)的对数：

- $i<j$
- 定义G为全1矩阵，$|G+F_i*F_j|\neq 0$

$N\leq 60, B\leq 8000$

28.3 算法讨论

若c为全1列向量，那么$(F_i,c)*(F_j,c)^T=G+F_i*F_j$。记$Q_i=(F_i,c)$，$Q_{ij}$为$Q_i$删去第$j$列后的结果，那么根据Cauchy-Binet公式，

$$|Q_i*{Q_j}^T|=\sum_{k=1}^{N+1}\left|Q_{ik}*Qjk^T\right|=\sum_{k=1}^{N+1}|Q_{ik}|*\left|Qjk^T\right|$$

根据如上结论，对于一个矩阵，只需要记录一个长度为$N+1$的二进制数p_i，其中$p_{ik}=|Q_{ik}|$，最后枚举两个矩阵，判断$p_i\ xor\ p_j$的二进制位中1的奇偶性就能得出答案了。

接下来的问题就是如何快速求出p_i。注意到如果直接对F_i进行高斯消元，就可以得出$p_{i,N+1}$，由于高斯消元时只是对列操作，那么可以在进行消元时顺便将c加入进去一起消元，即消元之后的矩阵为Q'_i，那么容易发现$Q_{ik}=Q'_{ik}$，而由于消元之后的矩阵（不包括c那一列）是对角矩阵，所以删除第k列之后只需判断c'_k是否为1即可。

28.4 时空复杂度

时间复杂度：$\mathcal{O}\left(BN^3+B^2\right)$
空间复杂度：$\mathcal{O}\left(B+N^2\right)$

29 STREETTA

29.1 试题来源

Codechef MARCH 14 The Street

29.2 题目大意

给出一个序列，支持以下操作:

- 输入L, R, a, b，给区间L, R的值加上一个首项为a，公差为b的等差数列
- 输入L, R, a, b，添加一条过点(L, a)斜率为b的线段，线段右端点在$(R, a+(R-L)*b)$
- 输入t，询问与$x=t$相交的线段的交点中纵坐标最大值加上点t的值。

29.3 算法讨论

第一个操作可以用线段树直接维护。

对于第二个操作，考虑在线段树的每一个节点上维护一条线段，加入一条线段后分为以下三种情况（设新加入的线段为S_{new}，原线段为S_{old}）:

- 不存在原线段，则令$S_{old}=S_{new}$
- 其中一条线段在此区间内完全高于另一条线段，则保留较高的线段。
- 两线段相交，记区间中点为mid，不妨设交点在中点左侧，则中点右侧的位置可以视作情况2，那么保留右侧较高的线段，并将另一条线段传向左侧，递归执行这个过程即可。

询问时只需要把在线段树上经过的所有线段在t点的取值求最大值即可。

29.4 时空复杂度

时间复杂度：$\mathcal{O}\left(Nlog^2N\right)$
空间复杂度：$\mathcal{O}\left(N\right)$

30 CNTHEX

30.1 试题来源

codechef SEPT11 Counting Hexagons

30.2 题目大意

在$[1,n]$中选择六个数字，重复的数字不超过k个，其中最大的一个不小于L，其余木棍不超过X，求这些数字能构成一个六边形的方案数。

$n \leq 10^9, n-L \leq 100, X < L$

30.3 算法讨论

考虑六个数字能构成一个六边形的条件，即最大的数字大于其余数字和。那么由于最大的数字与其它数字的取值区间不同，所以可以直接枚举最大的数字x，接下来问题就转化为：求$x_1 \dots x_5$，使得$x_1 \leq x_2 \leq x_3 \leq x_4 \leq x_5 \leq X$，且$\sum_{i=1}^{5} x_i > x$的方案数。

由于只有五个数，所以可以考虑用数位dp的方法。记$f[i][j][k][c][u]$，表示从高到低做到了二进制的第i位，五个数字在这一位的值为j（j为一个5位的二进制数），五个数的大小关系为k（这里需要保证五个数按照升序排列，所以k表示某个数字与它之前的那个数字是否相等），下一位的进位为c，当前最大数与X的关系，以及当前和与x的关系均存在s中。这样，枚举下一位的各个数的情况，以及进位，即可转移。

30.4 时空复杂度

时间复杂度：$\mathcal{O}\left((n-L)*2^{20}\right)$

空间复杂度：$\mathcal{O}\left(2^{20}\right)$

31 FIBTREE

31.1 试题来源

Codechef SEPT 14 Fibonacci Numbers on Tree

31.2 题目大意

一个N个点的树，支持以下操作。

- A x y:将x到y路径上第k个结点的权值增加F_k（F_k为斐波那契数列第k项）
- QS x y:视x为根，询问以y结点为根的子树中所有结点的权值和
- QC x y:询问x到y路径上所有结点的权值和。
- R x:将所有结点的权值还原到第x个操作后的状态。

31.3 算法讨论

由于广义斐波那契数列（只需满足递推式）支持以下操作:

- 求值：由数列前两项求出任意项的值。
- 合并：即两个广义斐波那契数列a_i, b_i，由于递推式相同，新的数列仍然满足递推式。
- 求和：广义斐波那契数列，可以由前两项的值求出数列的和。（公式同斐波那契数列）

故可以用线段树来维护路径上的增加以及求和操作。

同时维护链以及路径信息，可以使用树链剖分来维护。由于线段树支持可持久化，故可以可持久化树链剖分。

31.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N * log^2 N\right)$
空间复杂度：$\mathcal{O}\left(N * log^2 N\right)$

32 TICKETS

32.1 试题来源

codechef MAY 12 Selling Tickets

32.2 题目大意

给出一个二分图，点数为$m+n$，每个左部的点出度均为2，求最大的正整数k，使得任意大小为k的左部点的集合，都存在一个左部的完备匹配。

$n \leq 200, m \leq 500$

32.3 算法讨论

由于每个左部的点度均为2，那么可以将左部的点以及两条临边缩成一条边。这样就行成了一张无向图。那么只要找到一个边集E，使得与这个边集关联的点集V满足$|E| > |V|$，那么最终答案就是$|E|-1$。可以发现，$|E| = |V|+1$，否则直接删去一条边仍然满足这个性质。而且不存在度为1的点，否则可以删去这个点以及这个点的临边。

于是，就只有可能是某个点度数为4，或者某两个点度数为3，其他的点度数均为2。那么，图的结构就有三种可能:

- 两个环之间有一条路径相连
- 两个环在顶点处相交
- 两个度为三的点之间有三条不相交简单路径

注意到重复经过一条边肯定不优，所以暴力dfs判断这三种情况即可。

32.4 时空复杂度

时间复杂度：$\mathcal{O}\left(m^3\right)$。
空间复杂度：$\mathcal{O}\left(m^2\right)$

33 QRECT

33.1 试题来源

Codechef SEPT 14 Rectangle Query

33.2 题目大意

给定一个二维笛卡尔平面，你需要支持下类三种操作:

- I x1 y1 x2 y2 插入一个左下角在(x1,y1),右上角在(x2, y2)的矩形。
- D index:删除第index条插入的的矩形。
- Q x1 y1 x2 y2:询问有多少之前加入的矩形，左下角在(x1,y1)，右上角在(x2,y2)的矩形有至少一个公共点。

33.3 算法讨论

注意到，直接统计有多少个矩形与其相交较为麻烦，所以可以考虑容斥原理。考虑两个矩形不相交的情况。若两个矩形不相交，那么肯定有一条直线（平行与坐标轴）将二者分开。但是，当一个矩形在另一个的左上角时，直接这样统计会将其计算两遍，所以，应该扣除这种情况，即$[x_1 < x_2]\&[y_1 < y_2]$。其余三个角落情况类似。

所以，我们要统计：$1.x1_i < X$的矩形个数。$2.x1_i < X, y1_i < Y$的矩形个数。对于第二种询问，可以用树套树或者分治来实现。

33.4 时空复杂度

时间复杂度：$\mathcal{O}\left(n * log^2 n\right)$
空间复杂度：$\mathcal{O}\left(n * log^2 n\right)$（树套树）或$\mathcal{O}\left(n\right)$（分治）

34 BWGAME

34.1 试题来源

Codechef APRIL 15 Black-white Board Game

34.2 题目大意

设一个排列P是合法的，当且仅当$\forall 1 \leq i \leq n, L_i \leq P_i \leq R_i$，求合法排列中，逆序对为奇数的排列个数与逆序对为偶数的排列个数的大小关系。

34.3 算法讨论

一个排列可以认为是一个全一行列式的某个展开项。而逆序对的奇偶性就对应着这个展开项的系数（即± 1），故此题等价与求一个行列式的值。这个行列式中每一行的元素是一段连续的1。

根据线性代数的知识，我们可以对这个行列式进行高斯消元。为了保证计算的方便，在计算过程中应保证所有元素仅取$\{0,1\}$。那么，我们可以把所有左端点在i的行放在一个堆H_i中，每次选取堆中右端点最小的一个元素$X=[l_x,r_x]$，那么对于这个堆中所有其他元素$Y[l_y,r_y]$，将其减掉X，得到新的元素$Y'=[r_x+1,r_y]$。也就是将堆H_i与H_{r_x+1}合并。

最后，计算出每一行留下的元素组成的排列的逆序对的奇偶性即可。

34.4 时空复杂度

时间复杂度：$\mathcal{O}(NlogN)$
空间复杂度：$\mathcal{O}(N)$

35 PARADE

35.1 试题来源

codechef SEP 12 Annual Parade

35.2 题目大意

给定一张N个点的图，你可以选择若干条路径进行游行。若路径不是环，需要支付C的费用；经过每一条边时需要支付费用；如果某个点没有被任意一条路径经过，需要额外支付C的费用。求最小费用，询问有多组，每组的C不同。

$N \leq 250, Q \leq 10000$

35.3 算法讨论

首先，考虑在C确定的情况下进行费用流建模。

将每个点i拆为两个$(i, i+N)$，分别表示入点与出点。然后对于原图中的边(u_i, v_i, w_i)可以直接连接一条连接$u_i, v_i + N$，流量为∞，费用为w_i的边。从起点向每一个入点连一条流量为1，费用为$-C$的边，从所有出点向汇连一条流量为1，费用为0的边。这样每一次增广就代表着多经过了一个点。

对于每一个询问做一次费用流不能通过此题。考虑到所有的增广得到的费用必为$w - C$的形式，其中w为走过的边的费用。那么就可以做一次网络流，记录下所有的w，然后对于每一次询问直接使用即可。

35.4 时空复杂度

时间复杂度：$mincostflow(2N, M)$

空间复杂度：$\mathcal{O}(N + M)$

36 SEAARC

36.1 试题来源

Codechef JUNE 14 Sereja and Arcs

36.2 题目大意

给定一个数组a_i，求满足以下条件的四元组(i,k,j,l)的数目。$a_i = a_j, a_k = a_l, i < k < j < l$

36.3 算法讨论

初看题目无从下手，可以考虑分类讨论。

记录数组cnt_s表示$a_i = s$的出现次数。对于$cnt_s < t$的情况，可以暴力枚举出所有$i < j, a_i = a_j$的情况，然后统计相交的对数即可。此处复杂度为$\mathcal{O}(Nt * logN)$

对于$cnt_s > t$，这样的s不会超过N/t个，那么只需要枚举s，然后考虑统计$a_i = a_j = s$的情况。此时可以扫描整个数组，然后对于每个$a_k \neq s$，统计$a_i = s, i < k$的i的个数c_{a_k}，对于每个$a_j = s$，统计$a_l \neq s, j < l$的l的数量$d_{a_l,j}$，那么我们枚举每一个a_j的时候，只需要求$\sum_k c_{a_k} * d_{a_l,j}$，即可。考虑$j$从左到右统计，那么只需要每次维护和式的变化量。此处复杂度为$\mathcal{O}\left(N^2/t\right)$

那么，取$t = \sqrt{\frac{N}{logN}}$时，可取到最优复杂度。

36.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N\sqrt{NlogN}\right)$
空间复杂度：$\mathcal{O}(N)$

37 TWOROADS

37.1 试题来源

codechef SEPT 13 Two Roads

37.2 题目大意

给定一些平面上的点，现在作两条直线，每个点的代价是这个点到最近的一条直线的距离的平方。求作两条直线使得所有点的代价和最小。

37.3 算法讨论

首先考虑只有一条直线的时候如何求最小值。记

$$w = \sum_{i=1}^{N} \frac{(kx_i + b - y_i)^2}{1 + k^2} = \frac{Ak^2 + Bb^2 + Ckb + Dk + Eb + F}{1 + k^2} \tag{1}$$

那么，

$$\frac{\partial w}{\partial b} = \frac{1}{1 + k^2} * (2Bb + Ck + E) = 0$$

即可得：

$$b = -\frac{Ck + E}{2B}$$

带入(1)，即可得：

$$w = \frac{A'k^2 + B'k + C'}{1 + k^2}$$

则：

$$\frac{\partial w}{\partial k} = 0 \Rightarrow k = \frac{(C' - A') \pm \sqrt{(C' - A')^2 + B'^2}}{-B'}$$

由此即可求出最小值。

下面考虑两条直线的情况。那么所有的点一定被两条直线的两条互相垂直角平分线划分为4个部分，每个部分属于某一条直线。那么，可以直接枚举这个划分（即先枚举一条角平分线，然后枚举另一条与之垂直的直线），然后计算答案即可。

37.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N^3\right)$
空间复杂度：$\mathcal{O}\left(N\right)$

38 TKCONVEX

38.1 试题来源

codechef JUNE 13 Two k-Convex Polygons

38.2 题目大意

给定n个棍子的长度和整数k，求能否在其中选出2k个棍子拼成两个凸多边形。使得两个凸多边形都恰好有k跟棍子组成,且任意相邻的边都不共线。

$k \leq 10$

38.3 算法讨论

类似三角形的判定，若最长边两倍小于周长，那么这些棍子就可以构成凸多边形。

注意到，如果只选择一个多边形，那么就可以先对棍子排序，然后选择相邻的k个棍子进行判定。由于$k \leq 10$，那么可以随机一个顺序，然后对于左右两部分分别进行判断，若两部分都存在一个凸多边形则结束算法。这样进行足够多次即可认为不存在合法方案。

由于任意k个棍子不能构成凸多边形的长度序列是指数级增长的（$k = 3$时就是斐波那契数列），故当棍子足够多的时候一定可以构成凸多边形。所以总的棍子数目不会太多，所以这个算法是可以通过的。

38.4 时空复杂度

时间复杂度：$\mathcal{O}(N * A)$ 其中A为参数。

空间复杂度：$\mathcal{O}(N)$

39 LECOINS

39.1 试题来源

codechef MARCH 13 Little Elephant and Colored Coins

39.2 题目大意

给出N种硬币，每个硬币有颜色c_i及价值v_i。Q次询问，每次需要回答使用这些硬币组成价值为S的钱时，最多能使用多少种颜色。

$N \leq 30, v_i \leq 2 * 10^5, S \leq 10^{18}, Q \leq 2 * 10^5$

39.3 算法讨论

注意到$v_i \leq 2 * 10^5$，另$V = min\{v_i\}$，记$F[i][j][k]$为，前i种颜色中使用了j中颜色，能够组成的$mod\ V = k$的最小的钱数是多少，那么每次考虑新加入一种硬币(v, c)时，$F[i][j][k] = min\{F[i-1][j][k], F[i-1][j-1][k-v]+v\}$，在使用的硬币以及颜色集合不改变的时候，还可以进行转移$F[i][j][k] = min\{F[i][j][k-v]+v, F[i][j][k]\}$，注意到这个转移构成了一些环，那么对于每个环只要跑完两次就能保证每个节点都被转移过了。

对于一个询问S，枚举答案ans，若$F[c][ans][S\ mod\ V] \leq S$，那么$ans$就可能成为答案。

可以使用滚动数组优化空间复杂度。

39.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N^2V + Q\right)$

空间复杂度：$\mathcal{O}\left(NV\right)$

40 COUNTARI

40.1 试题来源

codechef NOV 12 Arithmetic Progressions

40.2 题目大意

给定N个整数$a_1, a_2, ..., a_N$, 求有多少种按顺序选择三个数的方法使得他们构成一个等差数列。

$N \leq 10^5, a_i \leq 30000$

40.3 算法讨论

如果中间位置的数a_j确定，那么要求的就是$a_i + a_k = 2 * a_j, i < j < k$的对数，那么就是左右两边进行一次卷积。

那么可以对整个序列分块，块大小为t，在处理每一块的时候，记录这块之前的数的出现次数pre_i，以及之后的出现次数nxt_i，那么以下分为三类讨论。

- i, k存在一个块内，可以直接枚举i，然后将后缀次数加入答案$nxt_{2*a_j - a_i}$，同时更新后缀次数，前缀亦然，复杂度$\mathcal{O}\left(t^2\right)$
- i, k均不在块内，这时候可以对前缀后缀次数求卷积，然后直接统计答案即可。复杂度$\mathcal{O}\left(NlogN + t\right)$

  综上，并取$t = \sqrt{NlogN}$即可。

40.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N\sqrt{NlogN}\right)$
空间复杂度：$\mathcal{O}\left(N\right)$

41 CNTDSETS

41.1 试题来源

Codechef JAN 14 Counting D-sets

41.2 题目大意

求有多少点集，满足点集中的点都是N维整点且其直径恰好等于D。点集的直径是点集中最远的一对点的切比雪夫距离。两组点集被认为是相等的，如果两个点集之间可以相互平移得到。答案对M取模。

41.3 算法讨论

首先，点集直径等于D，可以转化为点集直径不大于D，那么只需要对$D, D-1$求出两个答案相减即可。

由于可以将点集平移，所以不妨设所有维度上的坐标范围都是$[0, D]$，且每一个维度都至少有一个点取到了0。那么，记恰好有i个维度没有取到0的方案数为$F[i]$，至少有i个维度没有取到0的方案数为$G[i]$，$G[i]$是方便求出的：

$$G[i] = 2^{(D+1)*(N-i)+D*i} * \binom{n}{i}$$

F和G的关系是：

$$G[i] = \sum_{j \geq i} F[i] * \binom{j}{i}$$

根据二项式反演：

$$F[i] = \sum_{j \geq i} G[j] * \binom{j}{i} * (-1)^{j-i}$$

$F[0]$即为答案。

41.4 时空复杂度

时间复杂度：$\mathcal{O}(NlogM)$
空间复杂度：$\mathcal{O}(N)$

42 DEVLOCK

42.1 试题来源

Codechef FEB 15 Devu and Locks

42.2 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M，允许前导0，答案对998244353取模

对于$1 \leq M \leq T$的所有M输出答案，$T \leq 15000$，$P \leq 16$，$N \leq 10^9$

42.3 算法讨论

很容易想到直接维护一个DP，记$F[i][j][k]$为有i位十进制数，对P取模后结果为j，数位和为k的方案数。那么，从$F[i][j][k]$转移到$F[i+1][j][k]$可以直接$\mathcal{O}(10 * P * T)$转移。

注意到，从$F[i][j][k]$转移到$F[2*i][j][k]$也是可行的，因为这就相当于对于前一半的i位数字，每一个的权重乘上10^i，而后一半不变，那么记$t = 10^i (mod P)$，转移就是$F[2*i][j][k] = \sum F[i][n][m] * F[i][j - n * t][k - m]$，这是一个卷积的形式，可以用$FFT$加速转移。

于是可以用类似快速幂的方法，用$logN$次转移得出答案。

42.4 时空复杂度

时间复杂度：$\mathcal{O}\left((P^2 * T + P * T * logT) * logN\right)$

空间复杂度：$\mathcal{O}(P * T)$

43 DIFTRIP

43.1 试题来源

codechef DEC 12 Different Trips

43.2 题目大意

给出一棵树，求有多少不同的路径(u,v)满足v是u的祖先。其中若两条路径长度相同，并且经过的点的度数依次相同，那么两条路经即为相同。

$N \leq 10^5$

43.3 算法讨论

对于每一个点，将其权值认为是度数，那么就相当于询问这棵树有多少不同的到某个祖先的子串。用类似后缀数组的思想，考虑倍增解决这个问题。记$F[i][j]$为在长度为2^j的子串中，以i节点出发的子串的排名。这样很容易直接倍增求解F数组。有了F数组以后，就可以构造出SA数据，接下来只需要快速求出lcp即可。

考虑$lcp(u,v)$，用类似倍增找LCA的方法，枚举一个i，每次判断$F[u][i] == F[v][i]$，若相等，那么$lcp(u,v) = lcp(father(u,2^i), father(v,2^i)) + 2^i$。这样就能在$log$的时间内求出两个串的$lcp$，不同子串个数也就可以统计了。

43.4 时空复杂度

时间复杂度：$\mathcal{O}(NlogN)$

空间复杂度：$\mathcal{O}(NlogN)$

44 MARTARTS

44.1 试题来源

codechef NOV 12 Martial Arts

44.2 题目大意

给出一个矩阵，每个位置有一个pair表示你用i队员，对手用j队员的时候你的得分以及对手的得分。现在，你先安排一个比赛顺序，然后对手可以选择一场比赛取消之。假设你的得分为H，对手得分为G，那么需要最大化$H-G$，同时最大化H，对手则反之。

$N \leq 100$

44.3 算法讨论

显然对手需要选择$H-G$最大的一场比赛取消。那么可以先对输入数据排序，然后从小到大枚举最大的一场比赛，当最大的一场比赛确定之后，剩下的就是求一个二分图最大匹配。直接求最大匹配会超时，考虑到枚举过程中相当于一条条加入新的边，就可以用KM算法实现动态加边。每次加边，需要将原来的匹配删除，然后重新进行一次增广，也就是会进行N^2次增广，每次增广复杂度为 $\mathcal{O}\left(N^2\right)$。

44.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N^4\right)$
空间复杂度：$\mathcal{O}\left(N^2\right)$

45 QTREE6

45.1 试题来源

Codechef DEC 13 Query on a tree VI

45.2 题目大意

给定一棵N个节点的树，每个节点有一个颜色(黑/白)，支持下列操作：

- 询问有多少点与点u连通。两个点u和v是连通的，当且仅当u到v最短路径上的所有点(包括u和v)颜色都相同。
- 切换点u的颜色(黑变白，白变黑)。

45.3 算法讨论

考虑将每个节点拆成两个点：黑点和白点，分别表示若当前节点为黑（白）时，以这个点为根的子树中同色联通块的大小。那么，每次询问就可以沿着链向上找到最浅的一个同色祖先，然后直接输出答案即可。对于修改，假设u点被改为了黑色，那么u的所有连续黑色祖先的黑点就都要加上点u的黑点的大小，同理u的所有白色祖先要减去u的白点大小。为了支持上述两种操作，可以用树链剖分或者动态树维护。

45.4 时空复杂度

时间复杂度：$\mathcal{O}\left(Nlog^2N\right)$
空间复杂度：$\mathcal{O}\left(N\right)$

46 LPARTY

46.1 试题来源

Codechef APRIL 15 Little Party

46.2 题目大意

给定M个有N个变量的集合，每个变量取值为$\{0,1\}$，你需要找出一些集合，使得这些集合覆盖的集合的并等于输入的集合，并且这些集合的大小总和最小。

若A是B的子集，那么称A覆盖B。

46.3 算法讨论

注意到$N \leq 5$，那么可以考虑搜索，接下来讨论剪枝策略。

首先，我们要求出可能成为覆盖的集合。注意到若$S \in T$，且S是合法的，那么T就不可能出现在答案中。可以证明这样得到的最终集合个数不超过$3^N/N$

我们可以对集合按照大小排序，然后进行最优性剪枝，同时维护这些集合的后缀覆盖并，即可进行可行性剪枝。可以使用位运算加速集合的关系判定。

加入以上剪枝即可通过此题。

46.4 时空复杂度

时间复杂度：搜索复杂度，上界$\mathcal{O}\left(2^{3^N/N}\right)$

空间复杂度：$\mathcal{O}(M)$

47 SHORT2

47.1 试题来源

codechef DEC 11 Short II

47.2 题目大意

给定p(一个质数)，问有多少对$a,b(a>p,b>p)$满足$(a-p)(b-p)|ab$。
$p \leq 10^{12}$

47.3 算法讨论

原题等价与求$a,b(a>0,b>0),ab|(a+p)(b+p)$的个数。即$ab|ap+bp+p^2$。以下分三种情况讨论：

$p \mid a,p \mid b$，此时相当于求$ab|a+b+1$的数对个数，一共有5对。

$p \nmid a,p \nmid b$，此时相当于求$ab|a+b+p$的数对个数，不妨设$a \leq b$，则$a \leq \sqrt{p+1}+1, b=\frac{a+p}{ka-1}$，令$d=ka-1$，那么应该满足$p \nmid a, d \mid a+p, a \mid d+1$。可以枚举$d,b$之中较小的一个，复杂度为$\mathcal{O}\left(\sqrt{p}\right)$。

$p \mid a,p \nmid b$或$p \nmid a,p \mid b$，不妨设$b \mid p$，那么$(a,b) \Leftrightarrow (a,\frac{p(a+p)}{b})$，所以这种情况的数对是第二种情况的两倍。

故，设第二种情况的答案为s，那么总答案就是$3s+5$。

47.4 时空复杂度

时间复杂度：$\mathcal{O}\left(\sqrt{P}\right)$
空间复杂度：$\mathcal{O}\left(1\right)$

48 MONOPLOY

48.1 试题来源

Codechef NOV 13 Gangsters of Treeland

48.2 题目大意

给定一棵树，一开始每个点都是独立的。每个点的权值定义为这个点到根路径上经过的不同独立块的个数，需要支持两个操作:

- 将点u到根路径上的所有点合并成一个新的独立块。即原来已经合并的会被拆散。
- 求一个子树的权值的平均值。

48.3 算法讨论

这个操作就是LCT的Access操作，所以可以直接模拟这个操作，均摊分析仍然成立。求平均值可以转化为求和。在每次合并两个点的时候，整棵子树内的所有答案都需要减一，这里可以用线段树维护dfs序支持子树加减。

LCT的总Access操作所更改的点不会超过$\mathcal{O}(NlogN)$个，所以总复杂度可以保证不超过$\mathcal{O}\left(Nlog^2N\right)$。

48.4 时空复杂度

时间复杂度：$\mathcal{O}\left(Nlog^2N\right)$
空间复杂度：$\mathcal{O}(N)$

49 FNCS

49.1 试题来源

Codechef NOV 14 Chef and Churu

49.2 题目大意

大厨有一个含N个数字的数组A，元素标号1到N，同时他也有N个函数，也标号1到N。第i个函数会返回数组中标号L[i]和R[i]之间的元素的和。大厨会对Churu进行一下两种询问：

- 将数组的第x个元素修改为y
- 询问标号在m和n之间的函数的值的和

49.3 算法讨论

此题可以用分块解决。

将函数分块，每一块记录数组中每一个位置在这一块的函数中被统计了多少次。那么修改一个位置的时候就可以直接更改这一块的和。而对于单个询问，可以用另一个分块维护，即对数组分块，每一块中记录块内前缀和以及整块的前缀和，这样询问复杂度可以降低到$\mathcal{O}(1)$。

49.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N\sqrt{N}\right)$

空间复杂度：$\mathcal{O}\left(N\sqrt{N}\right)$

50 MATCH

50.1 试题来源

codechef JUNE 12 Expected Maximum Matching

50.2 题目大意

给出一个二分图中每条边的存在概率，求二分图的最大匹配的期望值。二分图两部点数分别为N, M。

$N \leq 5, M \leq 100$

50.3 算法讨论

二分图的最大匹配可以用Hall定理求出。考虑暴力状压，要用2^N个状态表示一个点的子集是否满足连接的右部节点个数不小于左部。那么就需要2^{2^N}的状态表示。但是经过观察，大部分状态是不可能转移到的，打表发现有用状态只有不超过400个。所以可以预处理出所有状态以及状态之间的转移。

用$F[i][s]$表示右部做到第i个节点，左部子集的满足情况为s的概率，那么在右部枚举当前点连接的左部节点的情况$t \in [0, 2^N)$，然后转移给$F[i+1][trans(s,t)]$即可。

50.4 时空复杂度

时间复杂度：$\mathcal{O}\left(Ms * 2^N\right)$，其中$s$为有用状态数。

空间复杂度：$\mathcal{O}\left(Ms\right)$

51 QTREE

51.1 试题来源

codechef MAY 13 Queries on tree again!

51.2 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（简单图不包含重边和自环）。这个图中仅包含一个环，题目保证环的长度为奇数（即环包含奇数个节点）。支持以下操作：

- 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，找到一个连续的边的集合，使得集合中边的权值之和最大。

51.3 算法讨论

如果这个图是一棵树，那么可以使用树链剖分加线段树维护路径的最大连续子段和。由于整张图只有一个环，那么可以将环当做根，对于环以下的部分用熟练剖分维护，环上部分单独用一颗线段树维护。那么最终答案就是两段树上路径加上一段环上最短路。

51.4 时空复杂度

时间复杂度：$\mathcal{O}(NlogN)$
空间复杂度：$\mathcal{O}(N)$

52 RIN

52.1 试题来源

Codechef DEC 14 Course Selection

52.2 题目大意

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：对于所有的$i(1 \leq i \leq K)$,A[i]是B[i]的前置课程。

对于每项课程i和学期j。X[i][j]表示铃在学期j选修课程i所能得到的期望分数。

求所能得到期望分数的平均值的最大值。

52.3 算法讨论

考虑一条从源到汇的链，若求一个最小割，则有且仅有一条边被割。这就对应了一项课程在某一个学期被选。称第i门课第j个学期被选的边指向的点为V_{ij}

那么，先后关系就可以表示为，若A要先于B选择，那么不能出现割在B的边比A的边更靠前。也就是说，若$i < j$，则$s \to V_{Bi}, V_{Aj} \to t$不能同时联通，那么我们就连一条$V_{Bi}$到$V_{Ai-1}$的边权为$\infty$的边限制即可。

52.4 时空复杂度

$maxflow(N * M, N * M)$

53 GNUM

53.1 试题来源

Codechef JULY 14 Game of Numbers

53.2 题目大意

给定两个数组A_i, B_i，维护两个二元组的集合S1,S2，初始时集合均为空。每次操作，选择两个数对$(i,j),(p,q)$，满足$(i,j), A_i < B_j$不在S1中,$(p,q), B_p < A_q$不在S2中，且$gcd(A_q, B_p, A_i, B_j) \neq 1$。如果这样的数对存在，他会将$(i,j)$,$(p,q)$分别加入到集合S1,S2中。

求最多可以进行多少次操作。

53.3 算法讨论

我们可以枚举每个数对(i,j)，则显然数对可以根据A_i, B_j分为不相交的两类，那么这两类数对之间，最大公约数不为1的数对连边，要求的就是这张图的最大匹配数了。

由于只需要知道最大公约数是否为1，那么我们可以建立一些虚拟节点，每个节点表示一个质数。若数对的两个数均包含这个质数，那么这个数对就与这个质数连边。这样就可以把边数降为$N^2 * logW$的级别了。

53.4 时空复杂度

$maxflow(N^2, N^2 * logW)$

54 LEBOXES

54.1 试题来源

codechef MAY 12 Little Elephant and Boxes

54.2 题目大意

给出n个盒子以及m个物品，每个盒子有p_i的概率获得v_i的钱，否则获得一颗钻石。第i个物品需要c_i的钱以及d_i的钻石。求能买到最多物品的期望值。

$n, m, d_i \leq 30, c_i, v_i \leq 10^7$

54.3 算法讨论

首先求出在给定钱v以及钻石d的情况下最多能买多少物品。记$f[i][j]$为给出i个钻石，买j个物品需要的最小钱数，那么显然f数组是可以dp求解的。然后只需要枚举答案，判断$f[d][ans], v$的大小关系即可。

由于$n \leq 30$，故$2^{n/2}$的算法是可以接受的，考虑折半搜索。合并答案的时候，只需要枚举左右分别的到了多少钻石，就可以维护一个指针线性求出概率了。

54.4 时空复杂度

时间复杂度：$\mathcal{O}\left(n^2 * 2^{n/2} + n^2 m\right)$，其中$s$为有用状态数。

空间复杂度：$\mathcal{O}\left(2^{n/2} * n\right)$

55 SEINC

55.1 试题来源

Codechef MAY 14 Sereja and Subsegment Increasings

55.2 题目大意

给一个数组，每次你可以选择一段区间，将整段区间加一。当整个数组在模4意义下全部为0的时候结束。求最小操作步数。

55.3 算法讨论

首先，为了构造一个合法解，我们可以将整个数组改为不降的。也就是要求区间的右端点必须为N。

那么，在这个过程中，有些位置是可以由某些区间在这里开始改为某些区间在这里结束。也就是说将一些区间的右端点“分配”给这个位置。那么，对于本来要加3的位置，改为分配即可将答案减3，那么也就是首先保证这些位置尽量改为被“分配”，然后再考虑加2的位置，然后再考虑加1的位置。

要注意改为“分配”的时候应该满足整个解的合法性。

55.4 时空复杂度

时间复杂度：$\mathcal{O}(N)$
空间复杂度：$\mathcal{O}(N)$

56 MINESREV

56.1 试题来源

codechef JUNE 11 Minesweeper Reversed

56.2 题目大意

给出一个$R*C$的扫雷地图，现在你可以点击一个格子来“关闭”格子。当你点击格子c_1以后，若c_2和c_1在正常的扫雷下能通过点击c_1一起打开，那么它们就可以一起被关上。特别的，你在点击一个雷的时候，只能关闭这一个格子。求最少点击次数。

$R, C \leq 50$

56.3 算法讨论

首先，不与空白格子相邻的数字以及雷必然要用一次点击将其关闭，所以这一部分可以直接计算。

注意到这时整个棋盘被分为了一些联通块，由于点击一个格子最多打开两个联通块，所以可以将能同时打开的两个联通块之间连边，然后用带花树求出最大匹配即可。

56.4 时空复杂度

时间复杂度：$\mathcal{O}\left((RC)^3\right)$，由于棋盘形式的限制，这里有一个很小的常数因子。

空间复杂度：$\mathcal{O}\left(RC\right)$

57 TAPAIR

57.1 试题来源

Codechef JAN 14 Counting The Important Pairs

57.2 题目大意

给出一张图，询问有多少个无序点对(i,j)满足删除i,j两条边后，图不联通。

57.3 算法讨论

首先，随意构造一棵DFS生成树。接下来考虑三种情况:

- 删除两条非树边，此时原图仍然联通。
- 删除一条非树边以及一条树边，此时要求删除的树边仅被这条非树边包含。此处可以使用树上差分维护树边被包含的情况。
- 删除两条树边，此时要求包含两条树边的非树边集合是相同的，那么两条树边中间的部分就被割开。这里可以对于每一条非树边随机一个权值，然后对于树边统计包含这条边的权值和，然后对权值和排序以后统计。

综上，即可求出答案。

57.4 时空复杂度

时间复杂度：$\mathcal{O}(NlogN)$
空间复杂度：$\mathcal{O}(N)$

58 FINDSEQ

58.1 试题来源

codechef FEB 12 Find a Subsequence

58.2 题目大意

给你一个长度为N的数组$a_0, a_1 \dots a_{n-1}$和一个“12345”的排列。你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这个排列一样。

$N \leq 1000$

58.3 算法讨论

首先，可以先枚举两个数，表示第2,4大的数的位置，随后，每一个数的取值范围就已经确定了，那么根据这些数的位置关系可以贪心地选择这些位置，若可以选择出这些位置，那么就找到了这样的排列。

例如：考虑排列为“23154”，那么可以选择一个在2位置之后的，距离2最近的一个大小关系在2和4之间的数字，然后继续贪心选择1等。这一部分选择可以用可持久化线段树优化。

58.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N^2 logN\right)$
空间复杂度：$\mathcal{O}\left(NlogN\right)$

59 TREECNT2

59.1 试题来源

Codechef MARCH 15 Counting on a Tree

59.2 题目大意

给定一棵N个点的带权树，边权为w_i，求点对(u,v)的数量，满足(u,v)路径上的边权最大公约数等于1。支持M次修改边权操作。

$N \leq 10^5, M \leq 100, w_i \leq 10^6$

59.3 算法讨论

由莫比乌斯反演的知识，对于任意的$D \leq 10^6$，求出$D|gcd(path[u \rightarrow v])$的$(u,v)$点对数，即可得出答案。由于每条边$w_i$，只会对$D|w_i$的$D$的计算时有贡献，那么就可以维护$10^6$张图，每张图中只加入这张图的权值的倍数的边，然后求出在每一张图中算出联通的点对数（可以用并查集维护），最后用莫比乌斯反演即可得出答案。

59.4 时空复杂度

时间复杂度：$\mathcal{O}((N+M)\alpha * 2^s)$，其中$s$为权值范围内的不同素因子个数。

空间复杂度：$\mathcal{O}(N)$

60 CONNECT

60.1 试题来源

codechef APRIL 12 Find a special connected block

60.2 题目大意

给出一个$n*m$个网格，每个格子有一个颜色以及代价。选出一个联通块包含至少k个颜色的最小代价是多少。

$n, m \leq 15, k \leq 7$

60.3 算法讨论

如果颜色数等于k，那么这就是一个简单的斯坦纳树的变种问题。

如果颜色数很多，但是最终答案用到的颜色就只有k个，不妨将所有颜色随机应射到一个$[1, k]$的颜色区间上面。这样，如果最终答案的k种颜色映射到了不同的颜色上面就能得出正确答案。所以进行一次映射的正确概率应该为$\frac{k!}{k^k}$，那么进行足够多次（我进行了2000次）映射就能通过此题了。

60.4 时空复杂度

时间复杂度：$\mathcal{O}\left((3^k * nm + 2^k * nmlog(nm)) * c\right)$，其中$c$为参数，这里取$c = 2000$

空间复杂度：$\mathcal{O}\left(nm * 2^k\right)$

61 HAMILG

61.1 试题来源

Codechef JULY 15 A game on a graph

61.2 题目大意

两个玩家进行一个游戏：选择一个起始点放上棋子，轮流将棋子沿一条边移动，不能走到已经走过的节点，求有多少先手必败点。

61.3 算法讨论

若起点不在最大匹配上，那么先手必须将棋子移动到某个在最大匹配上的节点。（否则找到交错路，最大匹配可以+1）那么后手就可以沿匹配边移动。那么这时先手要么无路可走，要么将棋子再次移动到某个最大匹配的节点上（理由同上）。故只需判断一个点是否在最大匹配上即可。

任意图的最大匹配用带花树解决。

61.4 时空复杂度

时间复杂度：$\mathcal{O}\left(V^3\right)$
空间复杂度：$\mathcal{O}\left(V+E\right)$

62 PARSIN

62.1 试题来源

codechef OCT 11 Sine Partition Function

62.2 题目大意

定义

$$f(m,n,x)=\sum_{k_1+k_2+\cdots+k_m=n} sin(k_1x)sin(k_2x)\dots sin(k_mx)$$

其中$k_i>0,k_i\in\mathcal{Z}$
求$f(m,n,x)$
$n\le 10^9,m\le 30$

62.3 算法讨论

考虑递推

$$\begin{aligned}f(m,n,x)&=\sum_{k_1+k_2+\cdots+k_m=n}\prod_{j=1}^{m} sin(k_jx)\\&=\sum_{k_1+k_2+\cdots+k_m=n}\prod_{j=1}^{m-1} sin(k_jx)sin((k_m+1)x)+f(m-1,n-1,x)*sin(x)\end{aligned}$$

对上式使用三角恒等变换，可以得到如下递推式：

$$f(m,n,x)=f(m-1,n-1,x)*sin(x)+f(m,n-1,x)*2cos(x)-f(m,n-2,x)$$

于是，维护一个$2m$个值的向量然后使用$(2m)^2$的矩阵快速幂加速递推即可。

62.4 时空复杂度

时间复杂度：$\mathcal{O}\left(m^3logn\right)$
空间复杂度：$\mathcal{O}\left(m^2\right)$

63 CIELQUAK

63.1 试题来源

codechef MARCH 12 Ciel and Earthquake

63.2 题目大意

有一个$R*C$的网格，每条边断掉的概率都为p，求左上角和右下角联通的概率。

$R \leq 8, C \leq 10^{18}$

63.3 算法讨论

当C不大的时候，可以用插头dp逐列计算答案。对于每一个轮廓线，用最小表示法表示联通情况，然后枚举当前的两条边的状态即可。

当C很大的时候，记f_i为$C = i$时的答案，可以发现$\frac{f_i}{f_{i-1}}$是收敛的。所以直接用快速幂计算答案即可。

63.4 时空复杂度

时间复杂度：$\mathcal{O}\left(R^R * c\right)$，其中$c$代表收敛到指定精度时的列数，此处为40。合法状压状态数远远达不到R^R这个理论上界。

空间复杂度：$\mathcal{O}\left(R^R\right)$

64 SEAORD

64.1 试题来源

Codechef NOV 14 Sereja and Order

64.2 题目大意

有N个程序，两台电脑，每台电脑只有一个线程，一个程序不能同时在两台电脑上运行。给出每个程序在每台电脑上的运行时间，求完成所有程序的最小时间。

$N \leq 10^4$

64.3 算法讨论

注意到此题N较小，而对于最终答案，只需要保证每个程序在每台电脑上的运行时间的区间不交即可，故对于同一个结果，对应的方案数应该为阶乘级别的。所以只需要随机调整多次即可，或者可以使用模拟退火。

64.4 时空复杂度

时间复杂度：$\mathcal{O}(N * k)$，其中k为参数

空间复杂度：$\mathcal{O}(N)$

65 TSUBSTR

65.1 试题来源

codechef APRIL 12 Substrings on a Tree

65.2 题目大意

给出一棵n个节点树，树上每个节点有一个字符，定义这棵树的一个子串为某一个根到其后代的路径形成的字符串。求这棵树有多少不同子串，以及每次给出一个新的字母表顺序，问第k大子串是什么。

$n \leq 2.5 * 10^5$

65.3 算法讨论

可以发现这个子串的定义就是广义后缀自动机上子串的定义，所以可以对整棵树建一个广义后缀自动机，然后dp求出每个点出发的不同子串的方案数即可。

65.4 时空复杂度

时间复杂度：$\mathcal{O}(n * m)$

空间复杂度：$\mathcal{O}(n * m)$，其中m为字符集大小，此处为26。

66 DGCD

66.1 试题来源

codechef JULY 12 Dynamic GCD

66.2 题目大意

给定一棵N个点的树，你需要支持：1.路径上所有点权值加上一个数。2.求路径上所有点的权值的最大公约数。

$N, M \leq 5 * 10^4, w \leq 10^5$，其中$w$为权值范围。

66.3 算法讨论

由于gcd可以合并，故可以用树链剖分把这个问题转化为序列的修改以及查询。

根据$gcd(x, y) = gcd(x, x - y)$，那么$gcd(a_1, a_2, \ldots, a_n) = gcd(a_1, a_2 - a_1, a_3 - a_2, \ldots, a_n - a_{n-1})$。可以把这个序列做一个差分，那么区间加就变成了单点修改，用线段树维护区间gcd即可。

66.4 时空复杂度

时间复杂度：$\mathcal{O}\left(Nlog^2Nlogw\right)$

空间复杂度：$\mathcal{O}\left(N\right)$

67 SEAEQ

67.1 试题来源

Codechef JULY 14 Sereja and Equality

67.2 题目大意

称两个长度为n的数组A,B相似，如果两个数组的每个对应位置的相对排名相同。

对于两个排列P1,P2，定义函数F(P1,P2)等于满足P1[l...r]相似于P2[l...r]($1 \leq l \leq r \leq n$)，并且P1[l...r]包含不超过E个逆序对的数对(l,r)的数目。求对P1,P2取遍所有n个元素的排列F(P1,P2)的总和是多少。

67.3 算法讨论

由于两个数组相似即规定了排名，于是问题转化为求一个$[1,n]$的排列，其中逆序对数不超过E。这可以用DP加前缀和优化解决。

67.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N^3\right)$
空间复杂度：$\mathcal{O}\left(N^3\right)$

68 EVILBOOK

68.1 试题来源

codechef MARCH 12 Evil Book

68.2 题目大意

给出n个物品，获得每个物品需要代价w_i，收益v_i的钱，你可以花x的钱将一个物品的w_i, v_i一起减小1/3。求达到某一个收益的最小代价。

$n \leq 10$

68.3 算法讨论

n只有10，而且w_i, v_i会缩小成小数，所以可以考虑搜索。有如下结论可以帮助剪枝：只有在要选择一个物品的时候，才会进行减小操作；所以可以在选择物品的时候枚举减小操作的次数，并且加入最优性剪枝即可。

68.4 时空复杂度

时间复杂度：$\mathcal{O}(n^c)$，其中c代表搜索上界，为玄学。

空间复杂度：$\mathcal{O}(n)$

69 CBAL

69.1 试题来源

Codechef MAY 15 Chef and Balanced Strings

69.2 题目大意

给定一个字符串，询问在区间$[L,R]$中所有平衡子串的长度和。平衡子串的定义是，所有字母出现偶数次。强制在线。

69.3 算法讨论

所有字母出现偶数次可以理解为，左右端点的字母前缀和的出现次数奇偶性相同。

那么显然可以使用分块的方法维护，记$F[i][j]$为第i块到第j个位置的答案，预处理$\sqrt{N}$个数组，然后每次询问再暴力处理出边界的答案即可。

69.4 时空复杂度

时间复杂度：$\mathcal{O}\left((N+M)\sqrt{N}\right)$

空间复杂度：$\mathcal{O}\left(N\sqrt{N}\right)$

70 FLYDIST

70.1 试题来源

codechef FEB 12 Flight Distance

70.2 题目大意

给出n个点的图，你可以改变一条边的权值，代价为改变的值。要求最终的图满足对于两点u, v，它们之间的最短路就是边u, v的边权。

$n \leq 10$

70.3 算法讨论

对于每条边设置两个变量d^+, d^-，那么每条边的权值w就便成了$w + d^+ - d^-$，代价就是$d^+ + d^-$，这里限制$d^+, d^- \geq 0$。用g_{ij}表示i, j之间的最短路，那么所有的约束就可以写成一些线性约束，那么可以使用单纯形算法求解这个线性规划问题。

70.4 时空复杂度

时间复杂度：$\mathcal{O}\left(simplex(n^2, n^2)\right)$
空间复杂度：$\mathcal{O}\left(n^4\right)$

71 DIVIDEN

71.1 试题来源

Codechef DEC 14 Divide or die

71.2 题目大意

给出一个$N°$角，求尺规作$1°$角

71.3 算法讨论

首先考虑不用给定的角能做出的最小正整数角度数为多少。

首先，容易作出$30°$角。考虑$36°$角，由于$cos(36°) = \frac{\sqrt{5}+1}{4}$,而尺规作图是可以开平方的，所以可以做出$36°$角。于是$3°$角也就可以构造出来了。

结合输入的一个$N°$角，若$N \neq 0(mod\ 3)$，则可以作出$1°$角。

71.4 时空复杂度

时间复杂度：$\mathcal{O}(N)$
空间复杂度：$\mathcal{O}(N)$

72 QUERY

72.1 试题来源

codechef FEB 13 Observing the Tree

72.2 题目大意

给定一棵N个节点的树，支持以下操作:

- 给一段路径路径加上一个等差数列
- 询问路径和
- 将整棵树恢复到第X次修改后的状态

$N, M \leq 10^5$

72.3 算法讨论

区间加等差数列可以直接维护首相与公差，两个等差数列相加可以首相、公差分别相加然后作为一个新的等差数列。

那么就可以直接维护一个可持久化树链剖分解决这个问题。

72.4 时空复杂度

时间复杂度: $\mathcal{O}\left((Q+N)log^2N\right)$
空间复杂度: $\mathcal{O}\left((Q+N)log^2N\right)$

73 ANDOOR

73.1 试题来源

codechef JAN 13 A New Door

73.2 题目大意

给出一个矩形画布以及N个圆，求画布上的圆的并的周长是多少。
$N \leq 1000$

73.3 算法讨论

对于每个圆A，考虑计算这个圆露出来的部分有多少。枚举另一个圆（或者一条画布的直线）B，B可能会挡住A的一部分弧，这部分弧对应的圆的极角一定是一个连续的区间，那么求出所有的区间以后排序统计没有被任意一个圆覆盖的区间长度就是露出来的部分，然后相加即为答案。

73.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N^2 logN\right)$
空间复杂度：$\mathcal{O}\left(N\right)$

74 RANKA

74.1 试题来源

Codechef JAN 15 Ranka

74.2 题目大意

两个人在一张9*9的棋盘上下围棋，支持跳过一步，要求已经出现过的状态不能重复出现（状态包含当前谁是先手），求一个合法序列使两个人下了至少N步。

$N \leq 10^4$

74.3 算法讨论

此题可以直接构造一组解。事实上，整个棋盘一共有2^{82}个状态，去除不合法状态以后还是远大于要求的步数。构造是容易的，可根据第二个人的棋子数分类构造。

74.4 时空复杂度

时间复杂度：$\mathcal{O}(N)$

空间复杂度：$\mathcal{O}(1)$

75 RIVPILE

75.1 试题来源

codechef JULY 13 RIVPILE

75.2 题目大意

给定二维平面上的n个木桩以及m个圆盘，你可以在木桩上装上圆盘，代价为圆盘的价值，求你能从$y = 0$通过圆盘走到$y = W$的最小代价。

$n, m \leq 250$

75.3 算法讨论

可以对于每一个木桩，建立m个点，分别表示这个木桩使用某个圆盘时的状态，然后在所有$n * m$个状态中，若两个状态可以到达，那么之间连边。最后使用dijkstra算法求一边最短路即可。

75.4 时空复杂度

时间复杂度：$\mathcal{O}\left(nm^2log(nm^2)\right)$
空间复杂度：$\mathcal{O}\left(nm^2\right)$

76 TMP01

76.1 试题来源

codechef SEP 13 To Queue or not to Queue

76.2 题目大意

对于一个字符串，要求支持在末尾插入一个字符，在开头删除一个字符，同时询问不同子串的个数。

操作数不超过10^5

76.3 算法讨论

后缀树可以支持动态在结尾插入一个字符。在开头删除字符就相当于直接删除了一个最长的后缀，可以记录每个后缀插入时的位置，然后直接在后缀树上删除，需要注意合并度为1的祖先节点来保证复杂度。

76.4 时空复杂度

时间复杂度：$\mathcal{O}(n)$
空间复杂度：$\mathcal{O}(n)$

77 BAKE

77.1 试题来源

codechef OCT 11 The Baking Business

77.2 题目大意

给出一些信息，每条信息属于不相关的四个关键字，有一个权重，关键字可以分为三级（例如省市区）。你需要支持插入信息以及检索某些关键字下信息的权重和。

每个关键字的范围$s \leq 20$，所有等级关键字和$t \leq 6$，信息数量$n \leq 10^5$

77.3 算法讨论

由于每一维关键字均不大，所以可以用多维数组来暴力维护这些信息。

77.4 时空复杂度

时间复杂度：$\mathcal{O}\left(s^t + nt\right)$
空间复杂度：$\mathcal{O}\left(s^t\right)$

78 GERALD07

78.1 试题来源

Codechef MARCH 14 Chef and Graph Queries

78.2 题目大意

给出一些边，每次询问若将$[L,R]$区间的边添加到图中，会有多少联通块。

78.3 算法讨论

由于此处需要维护的信息（连通块个数）不能快速合并，而且支持离线询问。故考虑分块做法。

考虑莫队分块，那么只需要支持加入一条边，维护联通块个数即可。这一步可以用并查集实现。

78.4 时空复杂度

时间复杂度：$\mathcal{O}\left((N+M)\sqrt{M}*\alpha\right)$

空间复杂度：$\mathcal{O}(N)$

79 PRIMEDST

79.1 试题来源

codechef AUG 13 Prime Distance On Tree

79.2 题目大意

给定一棵树。如果我们在树中等概率地选取两个不同的点，求这两个点之间的距离是一个质数的概率。

79.3 算法讨论

统计路径信息可以想到用树分治统计。那么所有长度的路径的出现次数就可以用卷积统计出来（需要用FFT优化）。由于当前子树的最长路径不会超过子树大小，所以可以保证总复杂度。

79.4 时空复杂度

时间复杂度：$\mathcal{O}\left(Nlog^2N\right)$
空间复杂度：$\mathcal{O}\left(N\right)$

80 PUSHFLOW

80.1 试题来源

Codechef AUG 14 Push the Flow

80.2 题目大意

给出一棵仙人掌，支持修改边权，询问两点之间的最大流。

80.3 算法讨论

树上两点之间的最大流就等于两点间边权最小值。对于一个环，可以把环上的最小边删掉，然后整个环的边权加上这条最小边边权即可。对于有修改的情况，可以用线段树维护环，动态树维护删掉最小边以后形成的树。

80.4 时空复杂度

时间复杂度：$\mathcal{O}(N * logN)$
空间复杂度：$\mathcal{O}(N)$

81 XRQRS

81.1 试题来源

Codechef JAN 15 Xor Queries

81.2 题目大意

给定一个数组a_i，满足$a_i \leq W$，你需要支持以下操作:

- 末尾增加一个数
- $i \in [L, R]$，输出a_i xor x的最大值
- 删除末尾k个数
- 统计$i \in [L, R], a_i \leq x$的i的数量
- 求区间k大

81.3 算法讨论

Trie树经典问题。

81.4 时空复杂度

时间复杂度：$\mathcal{O}(NlogW)$
空间复杂度：$\mathcal{O}(NlogW)$

82 ROC

82.1 试题来源

codechef FEB 13 Room Corner

82.2 题目大意

给出一个$N * M$的房间，墙用字符|, –表示，拐角用字符+表示，保证墙形成一个环，求两点间沿着墙走的距离。

82.3 算法讨论

本题的难点在于构这个环。首先，可以找到最左边的同时最靠上的拐角，这个角一定是朝向右下方的。然后沿着墙dfs整张图即可。

82.4 时空复杂度

时间复杂度：$\mathcal{O}(NM)$
空间复杂度：$\mathcal{O}(NM)$

83 SIGFIB

83.1 试题来源

Codechef AUG 14 Team Sigma and Fibonacci

83.2 题目大意

求

$$\sum_{x+y+z=N} 6 * x * y * z * Fibo[x] * Fibo[y] * Fibo[z])$$

对M取模的结果。

83.3 算法讨论

由于斐波那契数列可以递推求解，故上式也应该是一个常系数线性递推。经验证，该式为十二阶递推，可以直接递推求解。

83.4 时空复杂度

时间复杂度：$\mathcal{O}\left(s^2 * logN\right)$
空间复杂度：$\mathcal{O}\left(s\right)$，其中$s = 12$

84 CARDSHUF

84.1 试题来源

codechef JAN 12 Card Shuffle

84.2 题目大意

给出一个标号为$[1,n]$的数组，你需要支持：将一段数组取出插入到另一段中，将一段数组翻转。求最终结果。

$N \leq 10^5$

84.3 算法讨论

可以用Splay维护整个数组。模拟以上操作即可。

84.4 时空复杂度

时间复杂度：$\mathcal{O}(NlogN)$

空间复杂度：$\mathcal{O}(N)$

85 TWOCOMP

85.1 试题来源

Codechef JUNE 14 Two Companies

85.2 题目大意

给定一棵树以及一些链，每条链属于两个集合中的其中一个，并且带有权值。求一些链，使得属于不同集合的链不交，且权值和最大。

85.3 算法讨论

我们可以用暴力处理每一对链是否相交，然后就可以网络流建图，求最小割即可。

85.4 时空复杂度

$maxflow(N^2, N^4)$

86 ANUDTQ

86.1 试题来源

Codechef MAY 14 Dynamic Trees and Queries

86.2 题目大意

维护一棵树，支持新加入一个叶子节点，删除一棵子树，子树增加一个值以及询问子树和。

86.3 算法讨论

可以用平衡树维护欧拉遍历序。

86.4 时空复杂度

时间复杂度：$\mathcal{O}(N * logN)$
空间复杂度：$\mathcal{O}(N)$

87 LYRC

87.1 试题来源

codechef AUG 13 Music & Lyrics

87.2 题目大意

给出一些单词以及一些句子，求每个单词在这些句子中出现了多少次。

87.3 算法讨论

对单词建立AC自动机，然后把句子拿上去跑一边即可。

87.4 时空复杂度

时间复杂度：$\mathcal{O}\left(\sum L_i\right)$
空间复杂度：$\mathcal{O}\left(\sum L_i\right)$

88 RNG

88.1 试题来源

Codechef MARCH 15 Random Number Generator

88.2 题目大意

常系数线性递推。

88.3 算法讨论

经典问题，使用Caylay-Hamilton定理以及多项式取模即可优化递推。

88.4 时空复杂度

时间复杂度：$\mathcal{O}(N * logN * logK)$
空间复杂度：$\mathcal{O}(N)$

89 CONPOIN

89.1 试题来源

Codechef JUNE 15 Connect Points

89.2 题目大意

极大平面图判定。

89.3 算法讨论

极大平面图应该满足$E = 3 * V - 6$，当且仅当所有的域均为三角形时满足这个关系。除此之外，需要先判定该图是否是平面图。

89.4 时空复杂度

均为$\mathcal{O}(N + M)$

90 DAGCH

90.1 试题来源

Codechef FEB 14 Graph Challenge

90.2 题目大意

给出一个有向图，每个点的$sdom(x)$。

90.3 算法讨论

这是Codechef MAY 15 Counting on a directed graph的简化版，做法与之相同。

90.4 时空复杂度

时间复杂度：$\mathcal{O}(NlogN)$
空间复杂度：$\mathcal{O}(N)$

Part II
Challenge

91 KALKI

91.1 试题来源

codechef DEC 14 Kali and Devtas

91.2 题目大意

给定二维欧氏平面内的N个点，你需要返回这些点的一个生成树，使得C_i的最大值最小。C_i的定义是：对于每个点，设在生成树中与其相邻的点中最远的点的距离为R，那么以该点为圆心，半径R以内的的点的C_i全部增加1（包括自身）。

$N \leq 400$

91.3 算法讨论

这一题与生成树的边长度紧密相关。如果一条边很长，那么从直觉上看这应该是不优的。故可以构造一棵欧几里得最小生成树最为基准答案。然后多次在构造过程中加入随机因素即可。

91.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N^2\right)$
空间复杂度：$\mathcal{O}\left(N\right)$

92 DELNMS

92.1 试题来源

codechef AUG 13 Deleting Numbers

92.2 题目大意

给出一个数组a，你可以选择这个数组中下标为等差数列的一个子序列（$a_v, a_{v+d}, a_{v+2d} \dots a_{v+td}$要求满足$v+(t+1)d > n$），要求这个子序列中的元素的值都相同，然后删去这个序列。问最终删除整个数组的最小步数。

$n \leq 10^5$

92.3 算法讨论

可以发现，这个删除的限制非常严格，而且数据是随机的，所以可以考虑不是很优的贪心。即选择一个出现次数最多的数，将剩下的数全部当做单个数删除，最后一步删除这个数。

当然这个过程是可以重复的，选择了最后删除的数x之后，x的最后一次出现位置之后就变成了一个单独的子问题，可以递归解决，在序列长度很小的时候可以考虑状压DP。

92.4 时空复杂度

时间复杂度：$\mathcal{O}(N)$
空间复杂度：$\mathcal{O}(N)$

93 CLOSEST

93.1 试题来源

codechef JUNE 12 Closest Points

93.2 题目大意

给出三维空间的n个点，m次询问，每次询问距离一个点最近的那个点的编号。

$n, m \leq 50000$

93.3 算法讨论

此题可以使用kd-tree得到最优解，但是出题人特意构造了卡kd-tree的数据。所以需要把所有的点坐标沿着某一个点随机旋转一定的角度以后再建树、查询。

93.4 时空复杂度

时间复杂度：$\mathcal{O}(m\sqrt{n} + nlogn)$
空间复杂度：$\mathcal{O}(n)$

94 EDSTGRID

94.1 试题来源

codechef OCT 13 Edit Distance on Grid

94.2 题目大意

有一个黑白棋盘，支持三种操作:

- 交换两个四相邻的格子，花费代价1
- 将一个白色格子染成黑色，花费代价C_2
- 将一个黑色格子染成白色，花费代价C_3

要求你用最少的代价将整个棋盘上的黑色格子变成一个4-联通块，保证棋盘随机，黑色格子出现概率为p，p在$[0.05, 0.1]$中均匀分布。C_2在$[5, 20]$中均匀分布，C_3在$[20, 40]$中均匀分布。

94.3 算法讨论

首先，注意到交换的代价明显较低。所以，应该尽量不进行后两种操作。

如果不进行后两种操作，只是移动黑色格子使其联通，那么很容易想到一个算法：首先选择一个中心点，然后重复以下过程：选择一个离这个中心点所在的4-联通块最近的点，将这个点移动到这个联通块边。若移动代价大于删除代价，那么就直接删除这个黑色节点就好了。

然而，这个算法的表现并不好，分析结果可以发现，由于黑色节点分布较为稀疏，所以该算法删除了大部分节点。针对这个数据的性质，就需要设计一种算法，使得最终的四连通块尽量“远”地分布在全部的棋盘上。

一个可行的方案是，我们不止选择一个”中心点”，而是选择多个中心分布在整个棋盘上，最后再把它们连接起来。如果实现地足够精细，可以得到不错的方案。

另一种思路是，我们注意到若最终图形确定，那么问题就变成了一个二分图匹配的模型。而最终形状可以由贪心构造或者调整法得到。不过由于匹配复杂度较高，这种方法并不能很好地实现。

94.4 时空复杂度

时间复杂度：$\mathcal{O}\left((p * n * m) * (n * m)\right)$
空间复杂度：$\mathcal{O}\left(n * m\right)$

95 CHAORNOT

95.1 试题来源

codechef JUNE 13 To challenge or not

95.2 题目大意

给出一个数组a_i，长度为n，你要在这个数组中选择尽量多的数使得这个数组中不存在长度为3的等差数列。

$n, a_i \leq 10^5$

95.3 算法讨论

先假设这个数组就是$a_i = i$，这时考虑如何构造一个解。可以发现如果把这个数组平均分为3份，然后删除中间的一整份，那么左右两边的两份就互不相干了，即不会有一个等差数列同时包含这两份中的数字。这样构造出来的解满足下列等式：

$$T(n) = 2T(\lfloor \frac{n}{3} \rfloor), n \geq 3$$
$$T(n) = 1, n < 3$$

解得

$$T(n) = n^{log_3 2}$$

这已经是一个很优的解了。

可以发现，按照这样递归选择，就相当于将整个序列分为了一些不相关的小段，在这些小段内部可以使用暴力枚举的方法选择，当然这样删除存在冗余，可以在整个过程结束后尝试将所有的被删除的节点加入这个数组中。

对于输入的数组，只需要模拟执行找到三等分点的位置即可。通过这样的方法可以在清橙上拿到最高分。

95.4 时空复杂度

时间复杂度：$\mathcal{O}\left(N + ans^2\right)$
空间复杂度：$\mathcal{O}\left(N\right)$

96 SIMGRAPH

96.1 试题来源

codechef APRIL 12 Similar Graphs

96.2 题目大意

给出两张无向图，求一个排列p_i，当(i,j)在第一张图上有边，p_i,p_j在第二张图上有边时，得分加一，求最大化得分。

96.3 算法讨论

可以发现这个问题比图同构问题更加复杂，因为限制条件更弱，搜索算法并不能很好地解决这个问题。考虑模拟退火，这里的代价的计算可以用bitset优化为$\mathcal{O}(1)$，一次调整的复杂度为$\mathcal{O}(n)$，可以在每一个温度下选择$\mathcal{O}(n)$对点对进行尝试，选择代价最优的一对进行调整，这样可以得到不错的解。

由于此题特殊的数据构造，所以初始解为$p_i=i$已经非常优了。可以不需要多次随机初始解。

96.4 时空复杂度

时间复杂度：$\mathcal{O}$(玄学)
空间复杂度：$\mathcal{O}\left(n^2\right)$

97 STEPAVE

97.1 试题来源

codechef NOV 11 Stepping Average

97.2 题目大意

给出一个长度为n的数组以及一个整数k，每次你可以选择两个元素a_i, a_j，将这两个元素删去，然后加入$\frac{a_i+a_j}{2}$，如此重复直到最后只剩下一个数x，你的代价为$|x-k|$。要求最小化代价。

$n \leq 10^3$

97.3 算法讨论

可以发现，每个数在最终答案中的贡献是2^{-t}形式其中t为这个数被合并了多少次。所以，最初合并的许多的整数在最终答案中的贡献非常小，几乎可以忽略不计，所以重点就在最后的几次合并上面。

对于最先合并的那些数，最简单的想法就是距离k比较“远”的数先进行合并。那么定义一个数x的“偏差”为$k-x$，用一个堆维护这个偏差，每次选择偏差最大的两个数进行合并。在最后剩下的数不多的时候，就可以使用搜索算法解决。

另一个想法是，考虑一个数x，如果它是最后合并的两个数之一，那么另一个数就要接近$2k-x$，如此迭代即可。对于x的选择，可以贪心进行，也可以随机。但是需要特别注意k很小或者很大的情况，这时候答案很容易变得很大。

97.4 时空复杂度

时间复杂度：$\mathcal{O}(nlogn)$

空间复杂度：$\mathcal{O}(n)$

98 CHEFPNT

98.1 试题来源

codechef OCT 14 Chef and Painting

98.2 题目大意

给定一个$n*m$的棋盘，上面每一格的颜色为黑或白，每次你可以选择一条极大的连续白色格子染为红色，求最小的染完所有白色格子的步数。

$n, m \leq 100$

98.3 算法讨论

很简单的一个想法就是每次选择一条最长的白色格子进行染色，但是这样很可能“切断”了其余的格子。也就是说除了自身的颜色，还需要考虑对后续的染色的影响。

那么可以在设计估价函数时考虑到这一因素，对于左右都是白色的格子的染色方案扣除一定的分数。然后每次选择一个得分最大方案的染色即可。

还有一种想法是当前的染色需要“兼容”其他的染色，也就是对于两个冲突的染色方案同时扣除一定的分数。

上述方案在某些特定的图中都会变得很劣，所以需要总和使用多种方案来保证解的稳定性。

98.4 时空复杂度

时间复杂度：$\mathcal{O}\left((nm)^2\right)$

空间复杂度：$\mathcal{O}\left(nm\right)$

99 SEAVEC

99.1 试题来源

codechef NOV 13 Sereja and Vectors

99.2 题目大意

给出n个D维向量v_i，以及一个D维限制向量vec，要求你选出一些向量w_i使得$\sum_{i=1}^{s} w_i \leq vec$，最大化$s/sum$，其中$sum = \sum_{i=1}^{D} \left| \sum_{j=1}^{s} w_i - vec_i \right| + 1$

$n * D \leq 10^5$

99.3 算法讨论

由于要求最大化的式子十分复杂，所以可以考虑固定一个值来化简这个式子。首先可以随机一个顺序插入向量，这样就得到了一个初始解，那么考虑确定了这个解的向量数目，要求最小化sum，那么就可以随机两个向量，将一条向量替换另一条。多次替换之后就能得到一个不劣的解了。

99.4 时空复杂度

时间复杂度：$\mathcal{O}(nD * t)$。其中t为参数。
空间复杂度：$\mathcal{O}(nD)$

100 SEAND2

100.1 试题来源

codechef JAN 15 Sereja and Number Division 2

100.2 题目大意

对于一个数字A，对其进行数位重拍，要求最小化$\sum_{i=1}^{n} A\%B_i$

$n \leq 100, A \leq 10^{1000}$)

100.3 算法讨论

由于A很大，所以可以近似认为$A\%B_i$是一个随机数，而且随意交换两位以后这个随机数并不满足新的值在其临域中，所以对于这个随机数，随机调整就近似于多次随机，是无效的。但是直接随机多次取最优值就已经可以通过此题了。

一个想法是，对于A的高位数字可以直接多次随机，对于低位数字，可以考虑穷举所有的排列取一个最优解，这样的效果十分不错。

100.4 时空复杂度

时间复杂度：$\mathcal{O}(logA * n)$
空间复杂度：$\mathcal{O}(logA * n)$