试题泛做表格

姓名：于志竟成

序号	试题编号	名称	题目大意	算法讨论	时空复杂度
001	Codeforces 351D	Jeff and Removing Periods	给定序列，每次询问给出一段区间。求按照如下方式消除区间的最小步数： 选择一写间隔相同的相同数将其删除并将剩下的数以任意顺序重排。	第一次消除后把所有数按顺序排好后后面每次都能消除一种数。这样一定能保证不劣。于是只需要讨论第一次操作能否消除一种数，也就是这一段中是否存在一种数间隔相同，再加经典的计算区间内有多少不同的数就行了。 这两个问题都能用离线用线段树或者在线用可持久化线段树做。	时间复杂度：$O((N+M)\lg N)$ 空间复杂度：$O(N+M)$（离线） 或$O(N\lg N)$（在线）
002	USACO Mar 08	Land Acquisition	一些数对(a_i,b_i)，将其分组，每组的代价为$\max(a_i)\times\max(b_i)$。求最小总代价。	对于一个数对(a_i,b_i)，假如存在另一个数对(a_j,b_j)满足$a_i\le a_j,b_i<b_j$，则可以不考虑(a_i,b_i)，这个很显然。可以证明，将数对按a_i排序连续地分组不劣。然后注意到排序后b_i也是单调的。然后写个动规方程后发现可用比较正常的斜率优化了。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N)$
003	Codeforces 305E	Playing with String	有一堆字符串（初始时一个）两个人轮流进行如下操作： 选择一个串S和满足$S_{i\ 1}=S_{i+1}$的i，然后将串沿i两侧拆成三个串。 不能行动者输。问先手是否必胜。	容易想到一个字符不满足$S_{i\ 1}=S_{i+1}$时它后面也不可能满足，并且每次拆后仅有两头会由满足变为不满足。因此每段连续的满足条件的字符是独立的游戏，且我们只需要知道其长度。用 SG 函数就行了。	时间复杂度：$O(N^2)$ 空间复杂度：$O(N)$
004	Codeforces 277D	Google Code Jam	共有N道题目和时间限制L。每道题分难易两个子问题。给出解决每个简单子问题所需的时间和得到的分值，以及尝试解决每个困难子问题所需的额外时间、此题通过后得到的分值和此题 FST 的概率。罚时指最晚的正确提交的时间。求最大的期望得分以及在最大期望得分前提下的最小期望罚时。	期望得分与做题的顺序无关。我们尝试确定一种顺序，使得期望罚时尽量小。推一下可以发现这样的比较函数。剩下的就是比较简单的 DP 了。	时间复杂度：$O(N\lg N+NL)$ 空间复杂度：$O(NL)$
005	Codeforces 314E	Sereja and Squares	将每对字母看做一种括号。给出一个序列，其中有的位置是未知的，并且已知的位置只能是左括号。求在满足序列合法的情况下未知部分有多少种。	$O(N^2)$的 DP 是很容易想到的。但关键是10^5的数据范围让人不敢写。需要加些常数优化： 1.将数组从二维优化为一维 2.注意到每个位置左括号的奇偶性是确定的 3.不管明显废掉的状态（左括号多于一半）	时间复杂度：$O(N^2)$ 空间复杂度：$O(N)$

006	Codeforces 238D	Tape Programming	给定一种语言。这种语言每次向一个方向移动当前指针，然后根据当前指针所在位置执行操作，包括输出数、删除字符和改变指针移动方向。当指针指向序列外时终止。给一段序列，每次询问给出一段区间，问在这段区间中执行后各个数的输出次数。	如果从头部开始执行，我们发现，任意时刻已经到达过的位置都是一段前缀。因而对于询问的区间其执行过程都是从头部（或者从头部无法到达的另外一个位置）开始的执行郭晨过的一段。可以通过记录每一步执行的位置然后二分找到区间在其中的对应位置。通过记录输出的前缀和得到答案。模拟过程用并查集将删去的字符跳过即可。 不过，还有一个需要考虑的问题，那就是这样的执行步骤不会很多。注意到如果相邻两步都没有输出，则说明这两步都在箭头上，这样一定会删去一个箭头。所以步骤数还是 $O(N)$ 的。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N\lg N)$
007	Codeforces 235E	Number Challenge	给定 a,b,c 计算 $\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c}d(ijk)$，其中 $d(x)$ 表示 x 的约数个数。	$d(x)$ 显然为积性函数。也就是说我们只需要考虑各个质因子就行了。用 $f(i,j,k,p)$ 表示考虑了不比素数 p 更大的素数时，三个参数上界分别为 i,j,k 的答案。使用记忆化搜索即可。但要注意的是应当先考虑大的素数，这样有用状态会比从小的素数开始考虑少得多。	时间复杂度：$O(N^2)$ 空间复杂度：$O(N^2)$
008	Codeforces 306C	White, Black and White Again	给定 a,b,n，求将 a 个不同的 A 类元素和 b 个不同的 B 类元素分给 n 个不同的容器（容器内要分顺序，容器不能为空且每个容器只能装一种元素），并且要满足头部和尾部都是 A 类。	首先 A,B 中元素的排列方式有 $a!b!$，然后枚举有多少个容器用于装 A 类元素，将 B 类插在中间就行了。然后就是分堆，用隔板法就行了。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N)$
009	USACO Open 07	Connect	$2\times c$ 的网格支持断边、加边，询问一段两个端点的连通情况。	用线段树记录一段内 4 个端点的连通情况。	时间复杂度：$O(N+M\lg N)$ 空间复杂度：$O(N)$
010	Codeforces 264D	Colorful Stones	给两个只含 R, G, B 三种字符的串 A 和 B，长度分别为 N,M。一开始 $pA=pB=0$。假如给定一个也只含 R, G, B 的串 S，我们可以依次考虑 S 的每一项。假如 $A_{pA}=S_i$ 则令 pA 增加 1，假如 $B_{pB}=S_i$ 则令 pB 增加 1，注意这两个操作是同时完成的。给定的 S 需要保证任何时候 $pA<N,pB<M$。 问题是：给定 A,B，有多少个 (a,b) 满足存在 S 使	这里只把结论给出来： (a,b) 可达当且仅当 $\neg A[0,a]\leqslant B[0,b\quad 1]\wedge\neg B[0,b]\leqslant A[0,a\quad 1]$， 且满足 $a=0$ 或 $b=0$ 或 $A_a\neq B_{b\ 1}$ 或 $A_{a\ 1}\neq B_b$。 具体证明见专门的题解。	时间复杂度：$O(N+M)$ 空间复杂度：$O(N+M)$

			得最后 $pA=a$ 且 $pB=b$ 。		
011	Codeforces 317C	Balance	N 个容量为 V 的容器，每个容器制定初始水量和目标水量，某些容器间有双向管道相连。每一步可以将一个容器中的部分水通过管道运输至另一容器。任意时刻容器的水量不得超过其容量。求用不超过 $2N^2$ 步从初始状态转化至目标状态的方案。	各个连通分量显然是独立的，因此我们可以对每个连通分量分开考虑。首先如果初始水量之和与目标水量之和不相等，方案肯定不存在，这个显然。对于初始水量之和与目标水量之和相等的情况始终可以构造出不超过 $2N^2$ 步的方案。一个容器没有达到目标状态可以分两种情况：多了和少了。显然存在多了的容器当且仅当存在少了的容器。我们选择任意一对一个多一个少的容器，然后多的向少的输水。每次输水容易做到不超过 $2N$ 步。	时间复杂度：$O(N^2+M)$ 空间复杂度：$O(N)$
012	Codeforces 293E	Close Vertices	给一棵树，每条边有边权，给定路径的边数和边权和的限制，求符合限制的路径数目。	先进行点分治。然后问题转化成在平面上加一个点及求有多少点在左下角的一片矩形区域内，这就用分治加树状数组做。	时间复杂度：$O(N\lg^3 N)$ 空间复杂度：$O(N)$
013	Codeforces 264E	Roadside Trees	一个序列，有两种操作：删除从左到右第 K 个数和在某位置加入一个大小为 K 的数，$K\le 10$，并且每次操作后每个数会增加 1。求每次操作后序列的最长上升子序列。保证任意时刻没有相同大小的数。	对每次操作后数增加 1 我们只需要将数的大小加上一个标志时间的值就行了。注意到 K 是很小的。于是，如果我们记录以每个数开头的最长上升子序列的长度，当我们执行插入或删除、操作的时候，不超过 10 个数会受到影响。注意到这些数都是最小的或者最靠左的，于是我们只需要用两棵线段树维护一下 x 方向和 y 方向的值就行了。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N)$
014	Codeforces 325D	Reclamation	给一个左右相连的环形网格，每次尝试标记其中一个格子，要时刻保证存在一条连接顶部和底部的由未标记的格子组成的路径。问最终有多少个被标记的格子。	顶部和底部不连通当且仅当存在一个被标记的绕了一周的环。这个比较显然。然后问题就是如何快速判断是否存在这种情况。我们可以将环拆开，即在网格右侧拼一个相同的网格（新网格左右也是相连的），然后左侧网格中某个格子和右侧网格中某个格子通过被标记的格子连通就说明存在绕了一周的环。用并查集维护就行了。	时间复杂度：$O(\alpha N+RC)$ 空间复杂度：$O(RC)$
015	Codeforces 260E	Dividing Kingdom	给出一些平面上的点，要用两条不重合的水平直线和两条不重合的竖直直线将平面分成 9 个部分。给出这 9 个部分的点的数目，求任意方案。	首先 9!枚举 9 个部分的位置。然后通过 x,y 两个方向上的点数前缀和就能求出分割线的位置。然后再验证 9 个部分的点数（其实只用其中的 4 个部分）。验证的时候用可持久化线段树就行了。	时间复杂度：$O(4\times 9!\lg N)$ 空间复杂度：$O(N\lg N)$

016	Codeforces 283E	Cow Tennis Tournament	一些奶牛有各不相同的能力值。初始时能力之高的一定能够赢能力值低的。现在会将一些能力值范围内的比赛结果取反。求最后有多少 A 赢 B，B 赢 C，C 赢 A 的情况。	每个不符合条件的情况中都一定有一头奶牛同时赢另外两头奶牛。于是我们需要知道的就是每头奶牛能够赢多少头奶牛。我们可以将胜负关系写成 $N\times N$ 的矩阵，然后每次操作就变成了将矩阵中一个正方形区域内的结果取反。每次询问就是针对某一列。这可以用扫描线+线段树做。	时间复杂度：$O(N+M\lg N)$ 空间复杂度：$O(N+M)$
017	USACO Open 08	Cow Neighborhoods	给定平面上的一些点，两点之间连边当且仅当两点的曼哈顿距离不超过 C。求连通分量的个数和连通分量的最大点数。	先将曼哈顿距离转化为切比雪夫距离。考虑 BFS，每次从队首取点，然后找到所有没被访问且与其相邻的点，将这些点加入队列。问题转化成维护平面上的点，支持对某一矩形区域内的点的访问以及删点的操作。用线段树套平衡树就行了。但是这样常数较大。考虑到只有动态删点的操作所以可以用线段树套数组和并查集。	时间复杂度：$O(N\lg^2 N)$ 空间复杂度：$O(N\lg N)$
018	Codeforces 266E	More Queries to Array...	维护一个整数序列，支持区间赋值和询问 $\sum_{i=l}^{r} a_i(i-l+1)^k \bmod(10^9+7), 0\le k\le 5$	用二项式定理将询问的值展开，发现用线段树维护 $a_i i^k$ 的和就行了。	时间复杂度：$O(M\lg N+N)$ 空间复杂度：$O(N)$
019	Codeforces 323C	Two permutations	给两个 N 的排列，M 次询问有多少个数在第一个排列中的位置为 $[l_1,r_1]$，在第二个排列中的位置为 $[l_2,r_2]$，要求在线。	将每个数看成平面上的点，问题就成了求一个矩形区域的点数。 直接用可持久化线段树就行了。	时间复杂度：$O((N+M)\lg N)$ 空间复杂度：$O(N\lg N)$

020	USACO Dec 12	First!	给定一些字符串，问有哪些字符串可以通过指定一种字母表的顺序成为字典序最小。	对于两个串 $A<B$，则 A 和 B 第一个不同的字符的大小关系是确定的。因此建一棵 Trie 树，在对其 DFS 的过程中维护一个有向图，途中的每个节点即是字符。然后每次对这个图进行拓扑排序就行了。这里有一个优化：字符集大小仅为 26,（即图中的点数）因此可以用 32 位整型维护邻接矩阵。	时间复杂度：$O(26\times 26\times N+26\times L)$ 空间复杂度：$O(26L)$
021	Codeforces 286D	Tourists	给定一些区间及其出现时刻，多次询问若一个点在 t 时刻从原点以每单位时间 1 单位距离的速度向数轴正方向移动，其被至少一个区间包含的时间。	首先将可能相交的区间转化为不相交的区间。即计算分段后每段区间的最早出现时刻。区间变成不相交之后可以对每个区间独立考虑。然后容易发现使用树状数组能够维护各时刻出发受到的影响。	时间复杂度：$O((N+M)\lg M)$ 空间复杂度：$O(N+M)$
022	Codeforces 268D	Wall Bars	给定 $N\le 1000, h\le 30$，将 $1-N$ 范围内的整数分成四个不相交的递增序列（有编号），并且需要其中至少有一个序列满足存在 $S_i\ge N-h+1, \forall 0\le j<i: S_{j+1}-S_j\le h$。求合法的划分方案数。	h 很小于是想到很暴力的动态规划。状态需要记录四个序列中最近一个能够到尾步的数。这样复杂度为 $O(Nh^4)$。不加任何优化肯定是超时。不过注意到四个递增序列并没有任何区别，这样就很容易有效地减小常数。	时间复杂度：$O(Nh^4)$ 空间复杂度：$O(h^4)$
023	Codeforces 331C	The Great Julya Calendar	对一个非负整数 N，我们可以将其直接变为 $N-d$，其中 d 为 N 的十进制表示中的某个数字。求将 N 变为 0 的最小步数。$N\le 10^{18}$。	可以证明每次减掉尽可能大的数不会使答案变差。考虑将 N 分为两部分，即 $N=A\cdot 10^k+B$，则在 B 变为负数之前 A 中的最大数字都是可用的。因此，可以用 $f[i][j]$ 表示 A 部分中的最大数字为 i 且 $B=j$ 时将 B 变为负数所需的最小步数，$g[i][j]$ 表示变为负数后的值。转移很简单就不多说了。	时间复杂度：$O(100\lg N)$ 空间复杂度：$O(10\lg N)$
024	USACO Dec 10	Threatening Letter	给定字符串 A, B，每步可以选择 A 的任意子串将它们按顺序拼起来。求拼出 B 所需的最小步数。	显然假如拼出了 B 的前缀 i，我们下一步拼出后缀 $i+1$ 在 A 中以子串出现的最长前缀不会使答案变差。然后用后缀数组就行了。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N\lg N)$
025	Codeforces 342D	Xenia and Dominoes	一个 $3\times N$ 网格内有部分位置不可用，问有多少种用骨牌覆盖网格的方案，满足恰好有一个可用的位置未被覆盖（这个位置是给定的）并且这种覆盖方案能够移动一步（横放骨牌只能横向移动，竖放骨牌只能竖向移动）。	动态规划。$f[i][j][k]$ 表示考虑完前 i 个位置，轮廓线占用的状态为 $j(0\le j<8)$，当前能否移动为 $k(k=0,1)$ 时的方案数。转移和经典的骨牌覆盖区别不大。	时间复杂度：$O(N)$ 空间复杂度：$O(N)$

026	Codeforces 321D	Ciel and Flipboard	一个 $N\times N(N=2k-1,k>0)$ 的棋盘，每个位置有一个数。每次可以选择一个 $k\times k$ 的区域将其中的所有数取相反数。这样的操作可以执行任意次。求最后所有数的最大和。	用 $g[i][j]$ 表示一个数是否被取相反数。某个状态是可达的当且仅当对于每行的 $j<k$ 都有 $g[i][j]\oplus g[i][k]\oplus g[i][j+k]=0$，并且每列也满足类似的条件。然后只需要枚举中间的 k 个格子是否被取反，剩下的可以直接贪心了。	时间复杂度：$O(2^{(N+1)/2}N^2)$ 空间复杂度：$O(N^2)$
027	Codeforces 301C	Yaroslav and Algorithm	给定一些数，要求构造一串指令，使得对每个数执行这串指令后的结果都+1。每个指令指定将某个字符串变成另外一个字符串（可以包含?），并且决定接下来是终止还是继续运行。指令有优先级顺序，每轮迭代找到优先级最高的可以执行的指令执行。字符串替换时总是替换位置靠前的子串。	构造题。将?作为当前的指针位置。最开始我们需要插入一个?。考虑+1 其实是对字符串尾部进行一串修改，但是因为替换总是尽量考前只能先将指针放在头部。然后用更高优先级的指令将指针移动至尾部。这时需要将指针左移并进行一些修改操作。为了与之前右移的指针相区分，需要将?换成??。	时间复杂度：$O(1)$ 空间复杂度：$O(1)$
028	Codeforces 266D	BerDonalds	给一个无向连通图，边带有正权。找一个点使得图中节点到它的最大距离最小（这个点可以在边上）。	这个问题其实就是最小直径生成树问题。先用 Floyd 处理出每对点间的最短路，然后枚举点所在的边，设出这个点在这条边上的位置，然后其他每个节点到这个点的距离都是^形。需要计算这些^形取 max 最低的点。注意到这写最低点除开两边的话都是^的交点。将一些完整在其他^下的^删去后便都是相邻的^的交点。用单调栈就行了。	时间复杂度：$O(N^3)$ 空间复杂度：$O(N^2)$
029	USACO Open 09	Tower of Hay	给一个序列，要求将序列分成非空的段，使得段的和单调不增。求最大段数。	用 $f[i],g[i]$ 分别表示第 i 个数及之后的部分能够分出的最大段数以及在分出最大段数的条件下第一段的最小大小。转移用单调队列优化。	时间复杂度：$O(N)$ 空间复杂度：$O(N)$
030	Codeforces 263E	Rhombus	给一个 $N\times M$ 的表和整数 k，对于 $k\le x\le N-k+1,k\le y\le M-k+1$ 定义 $f(x,y)=\sum_{i=1}^{N}\sum_{j=1}^{M}a_{ij}\cdot\max(0,k-\lvert i-x\rvert-\lvert j-y\rvert)$。求使得 $f(x,y)$ 取得最大值的 (x,y)。	考虑枚举 x,y。注意到我们可以一格一格地移动 x,y，每次移动后一个三角形区域内所有数的权值都减一，另一个三角形区域内所有数的权值都加一。这个三角形区域的所有数之和可以用前缀和加减快速得到。初始的 f 值按定义暴力计算就行了。	时间复杂度：$O(NM)$ 空间复杂度：$O(NM)$
031	Codeforces 338D	GCD Table	一个 $N\times M$ 的表格，其中 i 行 j 列的值为 $\gcd(i,j)$。给定一个序列（长度为 K），问其是否在表格中的某一行出现。	先确定假如存在它应该在哪一行。首先这一行必须是 $\mathrm{lcm}(a_i)$ 的倍数。可以证明如果存在的话第 $\mathrm{lcm}(a_i)$ 行中也一定存在。因此我们就固定了行。现在将第一个列的位置设出来，然后可以得到一个同余方程组，用扩展欧几里得一步一步地将其解出来后带回验证即可。	时间复杂度：$O(K\lg a_i)$ 空间复杂度：$O(K)$
032	Codeforces 269D	Maximum Waterfall	给一些水平线段。水能从一条线段直接流到另一条线段当且仅当它们高度不同、水平上有相交部分且没有另外的线段挡在中间。它们之间水流量为水平相交的长度。水只能从高到低沿一条路流，不能分叉。总的水流量为中间每一步水流量的最小值，。求从顶部流到底部的最大水流量。	用 $f[i]$ 表示第 i 条线段流到底部的最大水流量。用平衡树维护水平坐标每个点最后被哪条线段覆盖，这样能够快速得到一条线段下面与之有交叉部分的线段。转移的时候还需要判断这两条线段之间是否有别的线段挡着。注意到这样转移的复杂度是有保障的，因为除了当前线段两个端点下的部分会被保留，中间的都会被直接删除。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N)$
033	Codeforces 317E	Princess and Her Shadow	一个无限大的网格图，有的格子有障碍物（障碍物都集中在一片小的范围内）。初始时 A 和 B 在两个	如果 A 和 B 都不在障碍物集中的区域内，我们便可以将 B 卡在某些障碍物处调整横纵坐标之差。如果它们在障碍物集中的区域内，我们可以让 A 沿着最短路追 B。这样在	时间复杂度：$O(N^2)$

			不同的无障碍物的格子上，要控制 A 上下左右移动。每步 B 也会尝试向相同方向移动，是否移动取决于要移动到的格子无障碍物。求一个方案使得最后 A 和 B 移动到了同一个格子上。	一定步数之后要么 A 和 B 已经在同一个格子中，要么它们已经到了障碍物集中区域外。	空间复杂度：$O(N^2)$
034	Codeforces 339E	Three Swaps	对一个排列我们每一步可以选择将某一段翻转。给定目标状态，求一个用不超过三步将 $1,2\cdots,N$ 变成目标状态的方案。保证方案存在。	由于最多只能执行三步操作，目标状态中只有不超过 14 位置个满足三个数不连续。这样的位置一定是某次翻转的端点。由于这样的位置数量很少我们暴力枚举每一步的操作就行了。	时间复杂度：$O(N\cdot 91^3)$ 空间复杂度：$O(N)$
035	Codeforces 325C	Monsters and Diamonds	共有 N 种怪兽，每种怪兽有多种分裂方式，每种分裂方式都是分裂成一些怪兽和至少一颗钻石。问从某种怪物开始分裂到只剩钻石时得到的最小和最大钻石数。	首先对于最小的问题，容易写出方程。注意到这个方程转移时会出现环，所以考虑用类似最短路的方式。这样可以得到永远无法分完的怪兽和分裂方式，将这些怪兽和分列方式去掉后做一遍 Tarjan，是某个点数大于 1 的强连通分量的前驱的怪兽可以分出无穷颗钻石，剩下的在 DAG 上 dp 就行了。	时间复杂度：$O(M\lg M+N)$ 空间复杂度：$O(N+M)$
036	Codeforces 238E	Meeting Her	给一个 N 个点的有向图，每条边权为 1。任何时刻有公交车从 s_i 到 t_i（共 M 种），公交车会随机选择一条最短路径。一个人能在公交车行驶中上车，也能在行驶途中下车。求从 A 到 B 最坏情况下所需乘车次数至少是多少。	用 $f[i][j]$ 表示乘第 i 种车在 j 点到 B 的最少乘车次数。方程容易写出。然后发现转移会出现环。使用类似最短路的方式解决之。	时间复杂度：$O(NM(\lg N+\lg M))$ 空间复杂度：$O(NM)$
037	GCJ 2009 Final C	Doubly-sorted Grid	给一个长方形网格，其中有些位置一定填入小写字母。问有多少种方案将其余位置填入小写字母使得网格每行从左至右、每列从上至下都是有序的。	注意到每种方案都唯一对应一组（26 个）从左下角至右上角的路径，满足只能向上或向右，并且相互之间不交叉。这种路径的个数为 $C(N+M,N)$，大约是 10^5。然后就能够用动态规划做了。	时间复杂度：$O(26\times N\times C(N+M,N))$ 空间复杂度：$O(NM+N\cdot C(N+M,N))$
038	Codeforces 309D	Tennis Rackets	有一个等边三角形，每条边上等距分布着 N 个点，并且与两个顶点最近的 M 个点不可用。现在在每条边上各选择一个可用的点，将其两两连接后要得到一个钝角三角形。求选点的方案数。	画个图推一下就能得到在 $O(1)$ 时间内检验方案的办法。然后 $O(N^3)$ 肯定是要超时的，注意到枚举了第一个点后，第三个点的可选范围会随第二个点的位置单调变化，然后就可以做到 $O(N^2)$ 了。但是要 AC 还需要做一些常数优化。主要有：有对称性只需要对第一个点枚举一半的位置、尽力用 int 而不用 long long、尽量用加减法代替乘除法、用一个变量代替两个变量。	时间复杂度：$O(N^2)$ 空间复杂度：$O(1)$
039	GCJ 2010 Final C	Candy Store	有 k 个顾客，每个顾客最多买 C 克糖果，现在需要用盒子来装糖果，每个盒子能装任意正整数克。问至少需要多少盒子才能应对任意情况。	结论题，感觉难做。用数学归纳法比较容易证明：从 $S=0$ 开始，若当前有 S 克糖果已装入盒中，且下一个盒子的大小设为 $[\frac{S}{k}]+1$，则每时每刻都满足如果 k 个顾客的购买总量不超过 S 时可以应对任意情况。然后，可以想到这样构造一定是最优的（下个盒子大了不可行，小了亏）。	时间复杂度：$O(\lg k+\lg C)$ 空间复杂度：$O(1)$

040	Codeforces 243C	Colorado Potato Beetle	一个无穷大的农田，初始时农夫站在某个格子，然后执行 N 条指令，每条指令是向某个方向走一定步数。在走过的格子都洒上杀虫剂。最后害虫会从四周无穷远处入侵，求有多少格子能够免受虫害。	农田很大，不可能硬搞。注意到关键的坐标（拐弯的位置）数量是 $O(N)$ 的，这说明我们可以将农田分成 $O(N^2)$ 个矩形区域，每个区域都可以当做整体对待。然后 FloodFill 一下就好了。	时间复杂度：$O(N^2)$ 空间复杂度：$O(N^2)$
041	Codeforces 360D	Levko and Sets	给两个整数数组 $A_1,A_2,\cdots,A_N,B_1,B_2,\cdots,B_M$，现在用它们生成 N 个集合。初始时第 i 个集合内仅有 1，然后每次选择集合内的一个数 c，将所有 $c\cdot A_i^{B_j} \bmod p$ 加入集合内，直到无法加入为止。问最终这些集合的并集大小。 p 是一个素数。	由于 p 是素数肯定存在一个模 p 的原根。假设这个原根为 r。对于第 i 个集合，其中的数形如 $r^{q_i\cdot k\gcd(B_j)} \bmod p$。由费马小定理，我们只需要关注 $q_i\cdot k\gcd(B_j)\bmod(p-1)$。它能够取得的值形如 $k'\gcd(q_i\gcd(B_j),p-1)$。我们要求 $\gcd(q_i\gcd(B_j),p-1)$，只需要找出最小的 $t\mid(p-1):A_i^{\gcd(B_j)t}\equiv 1(\bmod p)$，然后便有 $\gcd(q_i\gcd(B_j),p-1)=\frac{p-1}{t}$。最后转化成求一组数的倍数个数，用容斥原理就行了。注意这里容斥的时候不加记忆化会 TLE。	时间复杂度：$O(M\lg B+N\sqrt{p}\lg B+IE(N,p-1))$ 空间复杂度：$O(N+M+IE(N,p-1))$ （其中 $IE(N,p)$ 为 N 个 p 的约数进行容斥的复杂度。
042	Codeforces 273E	Dima and Game	一个游戏，初始时写下 N 对数 $(l_i,r_i)(1\le l_i<r_i\le M)$，两个人轮流选择其中的一对满足 $r_i-l_i>2$ 的数，将其变为 $(l_i+[\frac{r_i-l_i}{3}],l_i+2\times[\frac{r_i-l_i}{3}])$ 或 $(l_i,r_i-[\frac{r_i-l_i}{3}])$，不能操作的人输。问有多少种先手必胜的初始状态。	显然这 N 对数可以看做相互独立的游戏，只需要计算一对数的 SG 函数就行了。显然这个 SG 函数值只和 r_i-l_i 有关，我们用 $SG[x]$ 表示 r_i-l_i 的数对的 SG 函数值。由于每个状态的后继状态仅有两个，$SG[x]=0,1,2$。打表未能发现明显的规律，但是可以观察到随着 x 增大，连续的值相同的段的长度有指数级增长的趋势。于是我们可以考虑将这样的段看做计算的基本单位，用两个队列就能够很快地求出这样的段了。段的数目是 $\lg M$ 级别的，我们可以快速统计出 SG 值为 0,1,2 的数对个数。然后用一个简单 DP 就行了。	时间复杂度：$O(\lg M+N)$ 空间复杂度：$O(\lg M+N)$
043	GCJ 2009 Final A	Year of More Code Jam	一年有 N 天，共有 M 种线上赛，给定每种线上赛的每场比赛在第一场比赛进行后第几天进行。现在知道每种线上赛的第一场比赛的日期在这一年中等概率随机的一天进行。假设第 i 天有 x_i 场比赛，求 $\sum_{i=1}^{N}x_i^2$ 的期望值。	$E(\sum_{i=1}^{N}x_i^2)=\sum_{i=1}^{N}E(x_i^2)=\sum_{i=1}^{N}E((\sum_{j=1}^{M}d_{ij})^2)$ $=\sum_{i=1}^{N}E(\sum_{j=1}^{M}d_{ij}^2+2\sum_{1\le j<k\le M}d_{ij}d_{ik})=\sum_{i=1}^{N}(\sum_{j=1}^{M}E(d_{ij}^2)+2\sum_{1\le j<k\le M}^{M}E(d_{ij}d_{ik}))$，然后我们发现 $E(d_{ij}^2)$ 和 $E(d_{ij}d_{ik})$ 都很容易得到，于是我们可以快速算出一天的贡献了。注意到只有当 d 发生变化时一天的贡献才会变化，于是问题就解决了。	时间复杂度：$O(T\lg T)$ 空间复杂度：$O(T)$ 其中 T 为比赛总数。
044	Codeforces 335E	Counting Skyscrapers	一张大小为 N 的跳跃表，现在要从头走到尾，每次沿着高度为 x 的指针走便将 2^x 计入 $count$。这张跳跃表没有高度大于 h 的指针，求最终 $count$ 的期望值。	神奇的期望题。我们将高度限制逐渐放宽，然后每次便只需要考虑多加一层带来的贡献。这时我们考虑两个位置之间能得到更高的指针的概率，乘上带来的贡献就行了。这个贡献等于更高的指针的贡献减去覆盖的更低的指针的贡献。	时间复杂度：$O(Nh)$ 空间复杂度：$O(1)$

045	Codeforces 243D	Cubes	给一个 $N\times N$ 的正方形网格，每个格子中堆有 a_{ij} 个立方体。立方体的边长为 1×1。现在假设从无穷远处朝给定的一水平方向观察，求可见的不同立方体数量。	现在假定我们是向 $(x,y,0),x,y\geq0$ 方向观察，如果不满足我们可以将网格翻转。我们只需要考虑每个网格能看到的最低的立方体。考虑网格，过每个格子的两个顶点作视线的平行线，相邻的线之间被遮挡的状态总是相同的，所以如果我们由近及远依次考虑每个格子（近的才可能遮住远的），我们只需要用线段树维护平行线间区域的最小值就行了。	时间复杂度：$O(N^2\lg N)$ 空间复杂度：$O(N^2)$
046	GCJ 2009 Final E	Marbles	给 N 对数，以某种顺序排列，现在需要将每对数所代表的区间分为两个集合，使得每个集合的区间互不交叉，问两个集合重叠次数的和的最小值。	先构一张图，假如两个区间相互交叉，我们在它们之间连接一条边。显然有解当且仅当这张图为二分图。每个连通分量都可以看做整体，仅有两种安排的方案。并且，一定满足连通分量所代表的区间之间互不相交，即仅有包含和相离两种关系。然后用动态规划，$f[i][j]$ 表示标号为 i 的连通分量所代表的区间内第一个集合中重叠次数不超过 j 的情况下第二个集合重叠次数的最小值。由于没有交叉关系，连通分量之间的转移关系就成了树状的了，所以复杂度为 $O(N^2)$。	时间复杂度：$O(N^2)$ 空间复杂度：$O(N^2)$
047	Codeforces 301E	Yaroslav and Arrangements	定义满足相邻两项之差（包括头尾）的绝对值为 1 且第一项为最小项的序列为好序列。求有多少个长度不超过 N 且每一项为不超过 M 的正整数的单调不下降子序列满足通过重排可以得到大于零小于等于 K 个不同的好序列。	考虑动态规划。首先需要在状态中记下当前已经放入了多少个数以及应该考虑哪个值（即序列中的最大值），注意到任何一个可能通过重排形成好序列的方案一定都会唯一对应由最大的数形成几个“槽”，后面考虑的更大的值都应该填入这些“槽”中，而不同的重排方式就对应不同的填“槽”的方式。于是我们在状态中再计入槽的数量和填槽到当前状态的方案数。复杂度略高，但是加入一些状态上和转移上的常数优化就可以了。	时间复杂度：$O(N^4K)$ 空间复杂度：$O(N^3K)$
048	GCJ 2014 Final D	Paradox Sort	有 N 件物品，给定它们两两间比较的结果。如果我们有这些物品的一个排列，我们可以这样操作：按顺序考虑这些物品，如果当前手里没有物品或者考虑的物品比手中的物品更优我们就把考虑的物品拿在手上（如果手中有物品就把原来的物品扔了），否则不操作。最后我们会得到某个物品。求是否存在方案使得最后得到的物品为某一指定的物品，如果方案存在还要输出字典序最小的排列。	感觉很难。这里简单说下结论，具体证明见专门的题解。首先判断是否存在方案。如果在 A 优于 B 时连一条从 A 到 B 的边，那么方案存在当且仅当从 A 能够到达所有的点。然后字典序最小的方案就可以通过一位一位地贪心得到，但是判断部分方案是否存在略有不同，我们需要特殊考虑已经填入的这一段手中的最大值。详见题解。	时间复杂度：$O(N^4)$ 空间复杂度：$O(N^2)$
049	Codeforces 249D	Donkey and Stars	给定平面上的 N 个点，和 a,b,c,d。初始时 $A=O$，现在不断执行以下操作：从 A 向 $(b,a),(d,c)$ 两个方向分别作射线，这样形成了一个角，如果这个角内（不含边界）没有点则停止，否则任选其中的一个点作为新的 A。问最多能够执行多少次操作。	用 $f[i]$ 表示选了 $A=i$ 的最多操作次数，有 $f[i]=\max(f[j])+1$，其中 j 是从 i 射线得到的角中的点。很容易发现在角中等价于在一条直线的上方且在另一条直线的下方。于是就可以用排序和树状数组做了。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N)$

050	Codeforces 254D	Rats	一张 $N\times M$ 的网格图，其中有一些格子有障碍物，一些格子需要被覆盖。现在可以在两个不同的空格子内放炸弹，求一种 d 步后能够覆盖所有需要被覆盖的格子的方案。 $N,M\le 1000, d\le 8$	考虑取第一个需要被覆盖的点，由于它需要被覆盖，其中一个炸弹与它的距离不能超过 d 。由于 d 很小可以直接枚举第一个炸弹的位置。第二个炸弹也是类似的，随便选一个第一个炸弹没有覆盖的点然后枚举第二个炸弹的位置就行了。	时间复杂度： $O(NM+d^6)$ 空间复杂度： $O(NM)$
051	GCJ 2012 Final E	Shifting Paths	给一张 N 个点的有向图，其中 1 号点至 N 1 号点中的每个点都有且仅有两条出边，分别标号为 0 和 1，指向 t_{i0} 和 t_{i1} 。如果当前在 $i(1\le i<N)$ 号点，且之前访问它的次数为 c_i ，则必须沿着边 $t_{i(c_i \bmod 2)}$ 离开 i 号点。假如起点为 1 号点，问需要沿着边走多少次才能到达 N 号点，或指出永远无法到达 N 号点。	直接模拟复杂度为 $O(N\cdot 2^{\frac{N}{2}})$ ，肯定超时。考虑将点集 V 划分为 A,B 两个几乎等大的集合，我们处理出每个集合内的每个状态开始第一次到另一个集合中所需的步数和最终的状态。这样两个集合的状态数之和为 $N\cdot 2^{\frac{N}{2}}$ 。然后考虑用已经处理出来的这些信息来优化模拟。我们只需要关心在两个集合间的转移会发生多少次就行了。可以证明如果求出图中每个点到 N 号点的最短路长度，令 A 为距离 N 号点较近的那一半的点所构成的集合，则两集合间转移的次数为 $O(N\cdot 2^{\frac{N}{2}})$ 。具体证明见专门的题解。	时间复杂度： $O(N\cdot 2^{\frac{N}{2}})$ 空间复杂度： $O(N\cdot 2^{\frac{N}{2}})$
052	Codeforces 241B	Friends	给 N 个 10^9 以内的非负整数，求选择两个数异或的前 K 大的和。	构建一棵 Trie 树，然后我们就可以从高位到低位一位一位地决定第 K 大的值了。在这个过程中我们需要枚举两个数的其中一个，还需要处理 Trie 树的一棵子树内所有数与枚举的数的异或对和的贡献，我们维护子树内每一位为 1 的有多少个就行了。但是这样空间是 $O(N\ 31^2)$ 的，我们可以用 DFS 序的思路把空间优化到 $O(N\ 31)$ 。另外，清橙上给的时限常数卡得很紧。很关键的一个常数优化就是在中途不计算实际的贡献，仅统计每一位异或值为 1 的对数，最后一次性计算实际的和。这样可以大大减少乘法和取模运算。我在这样写之前怎么改都只有 75 分，改了后直接就 A 了。	时间复杂度： $O(N\ 31^2)$ 空间复杂度： $O(N\ 31)$
053	USACO Dec 07	Best Cow Line	给一个长度为 N 的字符串，现在需要用它构造一个新的长度为 N 的字符串，具体构造方法如下：选择开头或这末尾的字符，将其添加至新字符串尾部并将其从原串中删除，重复上述操作 N 次。问能够得到的字典序最小的新字符串是什么。	显然我们只需要比较当前的串正反两个方向哪个字典序更小。正着字典序更小就删去头部字符，否则删去尾部字符。比较字典序用哈希就可以了。	时间复杂度： $O(N\lg N)$ 空间复杂度： $O(N)$
054	USACO Open 13	Photo	有一个长度为 N 的序列，每一项为 0 或者 1。现在指定其中一些区间内所有项的和恰好为 1。求这个序列最大可能的和或者指出不存在合法的序列。	动态规划。首先，我们可以将给定的区间中出现的包含关系去掉。只要发现包含关系，我们就将大区间没有被小区间覆盖的部分全部打上标记表示它们必须为 0，然后只保留小区间。用 $f[i]$ 表示第 i 项为 1 时前 i 项最大的和。转移的时候我们需要找到包含这一项的最靠左的区间，然后用它的左边那个区间内的状态来更新。用线段树来优化转移。	时间复杂度： $O(N\lg N)$ 空间复杂度： $O(N)$

055	Codeforces 293D	Ksusha and Square	给定一个 N 个顶点的凸多边形，在其中（含边界）随机选择两个不同的整点，以它们间的连线为对角线作正方形。求正方形的期望面积。坐标范围 10^6。	我们要求的是 $E(dist(i,j)^2/2)=E(dist(i,j)^2)/2$。 $E(dist(i,j)^2)=E(x_i^2+x_j^2+y_i^2+y_j^2 \quad 2x_ix_j \quad 2y_iy_j)$ $=E(x_i^2+x_j^2)+E(y_i^2+y_j^2) \quad 2E(x_ix_j) \quad 2E(y_iy_j)$ ，发现只需要统计所有整点的坐标的和以及平方和了。我们找到凸多边形最左和最右的点然后枚举横坐标就可以统计出所有整点的信息。	时间复杂度：$O(N)$ 空间复杂度：$O(N)$
056	Codeforces 258	Little Elephant and Broken Sorting	给定一个 $1\sim N$ 的排列。先后有 M 个交换操作，每个操作交换指定的两项。但是每个操作被执行的概率均为 0.5，求最终序列的期望逆序对数。	我们用 $g[i][j]$ 表示一个排列的 i,j 两项是否为一对逆序对 $(1\leqslant i<j\leqslant N)$。则我们要求的是最终序列的 $E(\sum_{1\leqslant i<j\leqslant N} g[i][j])=\sum_{1\leqslant i<j\leqslant N} E(g[i][j])=\sum_{1\leqslant i<j\leqslant N}\Pr(R(i,j))$。于是我们只需要计算 $f[i][j]=\Pr(R(i,j))$，而每次交换只会改变 $O(N)$ 个 $f[i][j]$，于是问题就解决了。	时间复杂度：$O(N(N+M))$ 空间复杂度：$O(N^2)$
057	Codeforces 335D	Rectangle and Square	给定 N 个不重叠的矩形，求其中一个子集使得这个子集内的所有矩形恰好拼成一个正方形。坐标范围 $0\sim 3000$。	用类似于选四个点构成正方形的方法，我们枚举正方形的一个叫然后沿着点较少的一边暴力枚举。首先四个角必须满足条件，即这四个角必须同时是某个矩形的角。另外，正方形的边不能穿过矩形内部。满足这两个条件后，我们就能够确定有一些矩形被完整包含在这个正方形区域内了。由于没有重叠我们只需要用二维前缀和快速算出这片区域内所有矩形的面积和与正方形的面积比较就行了。	时间复杂度：$O(N^{1.5})$ 空间复杂度：$O(N)$
058	GCJ 2008 Final E	The Year of Code Jam	给定一个 $N\times M$ 的棋盘，我们需要在其中选择一些格子，有些格子可以选可以不选，有的必须选，有的不可选。每个选择的格子有一个贡献值，等于 4 减与其相邻的最多四个格子中被选的格子的个数。求最大贡献值之和。	先将棋盘黑白染色。然后，我们将白格子选、黑格子不选定看做 0，黑格子选、白格子不选看做 1。这样，任意两个相邻的数如果不同便会产生 2 的代价。然后就是个最小割的模型。	时间复杂度：$O((NM)^3)$ 空间复杂度：$O(NM)$
059	GCJ 2009 Final D	Wi-fi Towers	给定平面上的 N 个圆，每个圆都有给定的权值（可正可负）。我们要在这 N 个圆里面选择任意子集，满足对于一个在此子集内的圆，圆心在其内的所有圆也在此子集内。求选择的子集的最大权值和。	比较显然的最大权闭合子图问题，直接转化成最小割模型做。	时间复杂度：$O(N^4)$ 空间复杂度：$O(N^2)$
060	Codeforces 354D	Transferring Pyramid	给一个高度为 N 的金字塔，从上至下第 i 层有 i 个格子。我们现在需要覆盖一些给定的格子。有两种覆盖选择：覆盖单个格子，代价为 3，或者覆盖一个子金字塔，代价为金字塔格子个数加 2。不要求覆盖的格子是可以被覆盖的。求最小代价。	用 $f[i][j]$ 表示覆盖前 i 列的点，且当前子金字塔覆盖到最下面的 j 层的最小代价。然而这个状态数目是 $O(N^2)$ 的，显然不行。不过，注意到实际有意义的子金字塔的高度为 $O(\sqrt{N})$，所以仅需要计算 $O(N^{1.5})$ 的状态。转移可在常数时间完成。	时间复杂度：$O(N^{1.5})$ 空间复杂度：$O(N)$

061	Codeforces 333C	Lucky Tickets	求任意 M 个能够通过在数字间任意加入 +,−,× 以及括号得到 N 的长度为 8 的十进制数字串。	暴力枚举数字串的一半，计算出其能得到的值。然后考虑将两半简单和在一起。观察发现这样能够得到足够多的解。	时间复杂度：$O(M)$ 空间复杂度：$O(N)$
062	Codeforces 348E	Pilgrims	给一个棵 N 个点的树，边权非负。其中有 M 个点有标记。现在可以选择一个没有标记的点使其不能经过。问最多能使多少个带标记的点不能到达任何最远的带标记的点以及此前提下选择不能经过的点的方案数。	对于一个带标记的点，在其到所有最远的带标记的点的路径的交上选点可使答案+1，显然的是这些路径的交只会是连续的一段路径。最长路径的交可以通过两次 DFS+树形 DP 求出，然后用 DFS 序或树链剖分就可以得到最后每个点被覆盖的次数。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N)$
063	USACO Dec 05	Cow Patterns	给一个长度为 N 的文本串和一个长度为 M 的模式串。串 A 匹配串 B 当且仅当它们长度相同且任意两个位置在其中的大小关系相同。求模式能匹配文本串的哪些位置。字符集大小不超过 25。	从小到大处理每个出现的字符，如对任意的 k 第 k 小的字符出现的位置集合相同则两个串匹配。这样我们只需要用哈希维护和比较每种字符出现的位置集合就行了。	时间复杂度：$O(N+M)$ 空间复杂度：$O(N+M)$
064	Codeforces 305D	Olya and Graph	给一张 N 个点 M 条边的有向图，另外给一个正整数 k 求有多少种加边的方式使得最终图满足： 1.边都从小号的点指向大号的点 2.对于所有 $1\le j-i\le k, dist(i,j)=j-i$ 3.对于所有 $j-i>k, dist(i,j)=j-i\vee dist(i,j)=j-i-k$ 4.没有重边 其中 $dist(i,j)$ 为 i 到 j 的最短路长度。	显然，编号相邻的点之间一定有边，并且除这些边之外只能有跨越 k 个点的边，而且这些边必须两两交叉，这等价于其余边的起点必须再最左边的边跨越的 k 个点中。于是枚举最左边的那条边就行了。	时间复杂度：$O(N+M)$ 空间复杂度：$O(N)$
065	USACO Open 10	Triangle Counting	给定平面上的 N 个点，选择三个点作为三角形的顶点。求有多少个三角形包含原点。保证这 N 个点中没有任何一个点恰好是原点，也没有两个点间连线所在直线过原点。	显然，选出的三个点构成的三角形包含原点当且仅当过其中任何一个点与原点作直线后另外两个点在这条直线的异侧。直接统计不方便，考虑补集转化：计算不能包含原点的三角形个数。这时，一定有两个点在另外一个点与原点连成的直线的同侧。我们可以枚举一个点然后计算有多少点在其到原点的射线的左侧。将点按极角排序后扫一遍就行了。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N)$

066	Codeforces 309B	Context Advertising	给一段文字（由 N 个词组成）。现在要选其中连续的一段将其分成至多 r 行，并且要求每行里的词连续且词之间以空格相分割，且一个词不能被分到两行写，每一行的字符数（含空格）不能超过 c 个。求最多能够选由多少个词组成的段。	如果枚举了起点，每一行尽量向右延伸一定不劣。并且一行的终点位置随起点位置单调变化，故可以 $O(N)$ 扫一遍得到从每个位置开始一行能够覆盖哪些词。这样再枚举我们选的这段的起点就行了。但是朴素做法复杂度为 $O(Nr)$，我们需要用倍增优化，即处理出每个位置开始的 2^i 行能够覆盖到哪里。	时间复杂度：$O(N\lg r)$ 空间复杂度：$O(N\lg r)$
067	USACO Jan 07	Cow School	给 N 个有序对 $(t_i,p_i),0\le t_i\le p_i,p_i>0$，且满足对任意 $i\neq j,t_ip_j\neq t_jp_i$。求有哪些正整数 d 满足从中选择 $\frac{t_i}{p_i}$ 前 d 大的有序对得到的 $\frac{\sum t}{\sum p}$ 不是所有从中选择 d 个有序对的方案中最大的。	感觉这是很难做的题。设选前 d 大的答案为 k，则 k 不是最大的当且仅当存在一种方案 $\sum(t-kp)>0$。这等价于能够找到一个非前 d 大的有序对 (t_i,p_i) 和一个前 d 大的有序对 (t_j,p_j)，满足 $t_i-kp_i>t_j-kp_j$。显然，我们只需要分别找到这两部分中的最值就行了。简单暴力的方法就是用动态维护凸壳。时间复杂度 $O(N\lg N)$。但是这样编码复杂度较高，代码量较大。另有一种厉害的斜率优化算法，时间复杂度 $O(N)$，并且代码量很小。然而这种算法思维难度较大，较难理解。具体见另外写的题解。	时间复杂度：$O(N)$ 空间复杂度：$O(N)$
068	Codeforces 316D3	PE Lesson	有 N 个人，可以若干次选择两个人相互丢球。但是每个人都有丢球次数限制，最多只能丢一次或两次球。求最后有多少种不同的持球方式。	很重要的一个思路就是转换成有向图。每个点是一个人，有向边表示一个人手中的求到了另一个人的手中。显然，合法的有向图每个点的入度和出度都为 1，并且不同的图唯一对应了一种持球方式。这张图是由一堆简单环构成的。现在考虑一个环，只要其中只能丢一次球的人不超过两个就是可行的。然后就容易做了。	时间复杂度：$O(N)$ 空间复杂度：$O(1)$
069	Codeforces 319E	Ping-Pong	顶一个区间 S 能到区间 T 当且仅当 S 的两个端点中至少有一个在 T 中 。 依次执行 N 个操作，每个操作为加入一个区间或询问从一个区间能否到达另一个区间。保证加入的区间长严格递增。	首先可以转化成有向图的模型。考虑加入一个区间，已知这个区间的长度比之前存在的区间的长度都大。那么，包含其两个端点的所有区间一定和它交错，在图中就是一条双向的边。我们可以用并查集来维护这样的强连通子图。判断一个区间能否到达另一个区间的时候，首先判断它们是否在一个集合内，再判断能否通过单向的边到达。对每个集合我们都可以维护一个最左端点和最右端点，显然这两个端点间的每一个点都有区间覆盖。于是我们只需要判断 S 的端点是否在 T 所在集合的范围内就行了。需要用线段树来快速找到所有包含某个点的区间。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N\lg N)$
070	USACO Mar 09	Cleaning Up	给一个长度为 N 的序列，现在要将其分段，每一段的代价为其中不同的数的个数的平方。求最小的代价和。	用 $f[i]$ 表示将前 i 个数分段的最小代价。我们可以处理出每种数出现的上一个位置，然后代价仅有包含的这些位置不同才会变化。显然 $f[i]$ 单调不增，因此我们枚举了尾部覆盖的不同数的个数后可以贪心地更新当前的值。答案有一个显然的上界：N（分成 N 段），所以转移的时候不需要枚举大于 $\sqrt{N}$ 的部分。	时间复杂度：$O(N^{1.5})$ 空间复杂度：$O(N)$

071	GCJ 2011 Final A	Runs	对一个字符串，定义 run 为极大连续相同的子序列。现在给一个字符串，求将这个字符串重新排列后有多少个不同的字符串和原来的字符串有相同数量的 run。保证原串中 run 的数量不超过 100。	首先统计出可用的每种字符有多少个。然后，依次考虑加入每种字符。假设前面所有字符个数和为t，那么有$t+1$个不同位置可以插入新的字符。我们先枚举将当前这种字符分成多少 run，然后可以用组合数算出分段的方案数。然后将这些段填入$t+1$个不同位置中。如果填在原来的某一段 run 中间会使总 run 数额外加 1，因此我们还要枚举放多少个有分割作用的。最后还要用组合数算出选择插入的位置的方案数。容易想到在 DP 状态中记录当前已有多少 run 就行了。	时间复杂度：$O(N^3)$ 空间复杂度：$O(N)$
072	USACO Open 13	Figure Eight	给一个$N\times N$的网格，有的格子内有障碍物。现在要在其中找到上下两个空心矩形，满足上矩形的下边是下矩形的上边的矩形，并且边界不能越过障碍物。求两个矩形的面积之积的最大值。	考虑枚举中间的边（下矩形的上边界和上矩形的下边界），我们可以通过单调性在$O(N^3)$时间内求出中间的边的所有情况的上矩形和下矩形的面积最大值。对于上矩形的下边界需要是下矩形的上边界的子集，我们可以用二维前缀最小值来做。	时间复杂度：$O(N^3)$ 空间复杂度：$O(N^3)$
073	Codeforces 332D	Theft of Blueprints	给一个N个点的带权无向图，满足对于任意k个点，存在唯一的一个点与它们都直接连边，此时的代价为这些边的权值之和。求从图中选k个点的期望代价。	枚举与k个点都连边的点。即如果我们发现一个点与t个点连了边，那么共有$C(t,k)$种包含在内的方案。对于其中的每条边，其出现的次数为$C(t-1,k-1)$，故可以得到对其期望的贡献：$w_e C(t-1,k-1)/C(n,k)$。直接处理肯定有精度问题，将式子进一步化简就行了。	时间复杂度：$O(N^2)$ 空间复杂度：$O(N^2)$
074	Codeforces 280D	k-Maximum Subsequence Sum	维护一个长度为N的序列，支持单点修改和询问一段区间内取至多k个子段得到的最大和。	考虑网络流模型，其实就等价于反复进行“取出其中最大的子段，将其变为相反数”的操作。这样反复进行k次后取出的子段之和即为答案。所以我们只需要用线段树维护最大子段和就行了。	时间复杂度：$O(N+kM\lg N)$ 空间复杂度：$O(N)$ $O(N)$
075	USACO Nov 08	Toys	有N天需要用玩具。给定每天用的玩具数目和玩具价格（不随时间改变）。玩具用完后经过清洗可以重复用。有两种清洗方式，给定其用时和费用。问最小花费。	这题网络流的模型是比较明显的，但是肯定会超时。正解的思维难度就较大了。首先假如事先决定了我们要多少个玩具，我们是可以在一开始就把它们全部买好的，因为价格不随时间改变且这样不会使答案变差。然后，考虑维护一个清洗队列，为先后被清洗的玩具。队首为最早被清洗的玩具，队尾为最晚被清洗且已经可以用快洗投入使用的玩具。下面是贪心：假如队首的玩具可以用慢洗我们就用慢洗得到这个玩具，因为等再久能慢洗的玩具都不会减少；否则我们必须使用快洗，这时候显然应该用最晚被洗的玩具因为这能让玩具更可能变为可用慢洗的玩具。如果中间队列空了说明买的玩具不够。 现在考虑如何确定购买的玩具数量。可以证明，总花费关于购买的玩具数量的函数是个凸函数（两个凸函数之和）。因此可以用三分。	时间复杂度：$O(N\lg M)$ 空间复杂度：$O(N)$

076	Codeforces 248E	Piglet's Birthday	有 N 个架子，初始时每个架子上有给定数量的蜜罐 V_i（不超过 100）。现在要执行 N 个操作，每个操作是从一个指定的架子上随机选择 $k \le 5$ 个蜜罐，将其吃光，然后放到另一个指定的架子上。对每次操作输出执行操作后蜜罐被吃光的架子的期望个数。	这题中期望个数显然就等于各架子上没被吃的蜜罐数量为 0 的概率之和。想到用 $f[i][j]$ 表示第 i 个架子上有 j 个没被吃的蜜罐的概率。然后执行操作就是对 k 个罐子枚举其是否被吃过。注意这里 j 的范围，由于每次操作都不会使架子上没被吃的蜜罐的数量增加，所以 100 的范围就行了。另外，如果把 j 改为被吃的蜜罐的个数就会使问题变麻烦，因为被吃的蜜罐个数是可能增加的。	时间复杂度：$O((N+Mk)\max(V_i))$ 空间复杂度：$O(N\max(V_i))$
077	USACO Jan 09	Safe Travel	给一张无向带权图，满足从 1 号点到其他各点的最短路有且仅有一条。问每个点删掉原最短路最后一条边后从 1 号点到它的新的最短路长度。	由 1 号点到其余各点的最短路得到其最短路图一定为一棵树。现在要删掉树上的一条边，新的最短路一定是由一条不在树中的边连接一个子树和子树外部。比较容易发现直接将边按照 $dist[a]+dist[b]+len$ 排序后更新路径上的点的答案就行了。需要使用并查集以将之前已经更新过的点直接跳过。	时间复杂度：$O(M\lg M+N)$ 空间复杂度：$O(N+M)$
078	Codeforces 311C	Fetch the Treasure	有 h 个格子，其中有 N 个格子中有宝藏，给定它们的编号和宝藏价值。如果有一组数 r_i，那么可以到达编号为 $1+\sum a_i r_i, a_i \ge 0$ 的格子。初始时 r_i 仅有 1 个 k，且不超过 10000。接下来需要处理 M 个操作每个操作属于三个类型中的一个。 1.加一个 r_i 2.减小一个格子中宝藏的价 3.询问能够到达的格子中宝藏价值的最大值并将其中的宝藏全部取走。 保证操作 1 至多 20 次。	解决本题的关键是利用操作 1 至多 20 次且 $k \le 10000$ 的条件。考虑将所有格子按照模 k 的余数分成 k 类。注意到如果当前有个数 p 是可达的且在第 i 类中，那么大于等于 p 且在第 i 类中的所有数显然也都是可达的。因此可以用 $f[i]$ 表示第 i 类中最小的可达的数，这个可以用 dijkstra 求出，由于仅有 20 次操作 1，每次暴力更新就行了。最大值用堆维护就行了。另外，由于每次操作 1 后原来可达的数依然是可达的，因此不需要将堆整个重新建，只需要将新发现的可达的数加进堆就行了。	时间复杂度：$O(M\lg M+20k\lg k)$ 空间复杂度：$O(N+M+k)$
079	USACO Mar 13	Hill Walk	给定平面上的一组不相交的线段。现在从一个点出发，如果在一条线段上就一直向 x 轴正方向走到线段末端然后掉下去。重复以上过程直到掉到 y=-INF 处。求从第一根线段出发能够经过的线段数。	扫描线大法好。由于线段不相交，我们知道如果在某个 x 处线段 A 在线段 B 下面，那么在它们同时出现的地方 A 始终是在 B 下面。所以可以对所有端点横坐标排序，然后用平衡树维护线段的偏序关系。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N)$
080	USACO Mar 12	Cows in a Skyscraper	给定 N 头牛的重量和电梯载重 W，求最少需要坐多少次电梯。	暴搜+剪枝是可以过的。实际上可以用动态规划。$O(N^3)$ 的枚举子集转移就不多说了。 实际上可以达到更低的复杂度。定义当不能用 i 个电梯装下集合 j 时 $f[i][j]=0$，否则 $f[i][j]=sum[j]$。转移则为 $f[i+1][j]=(sum[j]-\max(f[i][j'])) \le W\ ?\ sum[j]:0$。 这里无需枚举子集，考虑每次加入一个头牛转移就行了。	时间复杂度：$O(N^2 2^N)$ 空间复杂度：$O(2^N)$

081	Codeforces 285E	Positions in Permutations	求 $1-N$ 的排列中有多少个个满足恰好有 k 个 $\lvert a_i-i\rvert=1$ 的位置。	先考虑确定那些满足 $\lvert a_i-i\rvert=1$ 的位置，其他的位置空着不管。那么，考虑到一个位置 i 时，我们只需要知道 i 和 $i-1$ 这两个数有没有被使用过，因为更小的数对后面没有影响，更大的数肯定还没有用。于是这样可以用 dp 求出一个方案数 $F[i]$。考虑其他位置的填充方式，有 $(N-i)!$ 种。注意到这里并不能保证剩下的 $N-i$ 个位置不满足条件。因此我们还需要用类似容斥的方法去除多余的情况。即假设已经计算出 $g[j](j>i)$ 为恰好 j 个合法位置的方案数，那么我们就可以将它们排除以计算 $g[i]$。	时间复杂度：$O(N^2)$ 空间复杂度：$O(N^2)$
082	Codeforces 295D	Greg and Caves	有一个 $N\times M$ 的网格，要将其中的每一个格子涂成黑色或者白色。现在求满足以下条件的方案数： 1.有连续的行满足每行恰有两个黑点，其余行全部是白点 2.这段连续的行满足向上或者向下走黑点间白色区域不扩大（始终是上一行的子集）	先只考虑一个方向，可以用 $O(NM)$ dp 求出方案数，然后考虑最高的不是上面的一行的子集的行，枚举其白条的长度和下部高度，然后可以用一个二维前缀和处理出上面部分对答案的贡献。	时间复杂度：$O(N^2)$ 空间复杂度：$O(N^2)$
083	USACO Jan 12	Cow Run	两个人博弈，每回合两个人先后都有两种选择。给定第二个人每回合的决策，保证第一个人一开始有必胜策略，求第一个人每一步的必胜策略。 回合数 $N\le 14$。	将所有状态列出，我们可以得到一棵树，这棵树的高度为 $2N$。如果在每个节点处记录该状态对于第一个人是否是必胜的，那么每一个节点的值是两个子节点值的或或者与，并且是按深度间隔变化的。这样我们就可以写个暴搜了。粗略估计其复杂度的上界为 $O(4^N)$，但是注意到与和或本质上没有区别，它们都是短路运算符，通过随机访问一个子树，可利用短路的这一性质将复杂度降至期望 $O((\frac{1+\sqrt{33}}{4})^{2N})$。	时间复杂度：$O((\frac{1+\sqrt{33}}{4})^{2N})$ 空间复杂度：$O(N)$
084	Codeforces 316G3	Good Substrings	给定一个字符串 S，求 S 中满足在给定的一组字符串中出现次数在给定范围（上下界之间）的不同子串的个数。	将 S 和给定的其他字符串以分隔符分隔拼成一个大串，对其建立后缀自动机，然后就可以得到每个节点对应的子串的出现位置，就可以知道是否满足给定的范围，是 S 的子串这一条件可以转化成在 S 中出现次数大于等于 1。建立 parent 树再 DFS 一遍，然后计算出每个节点代表的子串的数量就行了。	时间复杂度：$O(NM)$ 空间复杂度：$O(NM)$ N,M 分别为字符串总长度和个数。
085	Codeforces 235C	Cyclical Quest	给定一个串 S，多次询问，每次询问给定另外一个串 C，问 S 中有多少个子串（位置不同即算作不同）与 C 周期性同构。所谓周期性同构是指可以通过旋转相互得到。	建一个 S 的后缀自动机，处理出每个节点所代表的字符串的出现次数。然后将 C 复制一遍拼在后面。我们现在要做的就是在 S 的 SAM 中找到复制后的 C 中所有长度为 C 的长度的串。当然暴力是不行的。注意到每个串都可以由前一个串平移一个位置得到，也就是原串的一个后缀再加上一个字符。因此我们可以直接将复制后的 C 中的字符依次输入给 SAM，走不动时就走 parent 边，维护当前匹配的最长后缀的长度。当最长后缀大于等于 C 的长度时，我们就可以沿着 parent 找到对应的节点。当然还需去重，这个只需要把每次找到的节点打上标记就可以了。	时间复杂度：$O(N)$ 空间复杂度：$O(N)$ N 为所有字符串的长度之和。

086	Codeforces 241D	Numbers	给定一个 1 到 N 的排列，要选择一个非空子序列，满足其异或和为 0 且将其中的数依次拼成一个大的十进制数后模 p 为 0。p 保证是质数。	比较巧妙。一般的 DP 复杂度为 $O(N^2 p)$。需要尝试减小 N。注意到，当 N 达到 30 的时候可选子集的个数就非常多了，而我们可认为每个集合拼成的数在模 p 下大致是随机的，因而得到 0 的概率非常高，所以我们直接在排列中取出不超过 30 的部分就行了。	时间复杂度：$O(30^2 p)$ 空间复杂度：$O(30^2 p)$
087	GCJ 2009 Final B	Min Perimeter	给定平面上的 N 个点，求其中三角形的最小周长，退化的三角形也算。	类似于最近点对的分治算法。将点按照横坐标分成两半，那么最小的三角形一定要么整个在同一边，要么部分在左边，部分在右边。假设目前的最优答案为 ans，那么中间可能更新答案的三个点肯定在一个 $ans/2\times ans$ 的区域内，并且这区域的两半内的点都满足三角形的周长不小于 ans，可以想到其中的最大点数是很小的常数，故直接暴力更新就行了。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N)$
088	Codeforces 306D	Polygon	给定正整数 N，要求构造出一个 N 个点的凸多边形满足其所有内角大小相等，其所有边长两两不等。	$N<5$ 时显然是无解的。然后想了一下感觉如果有解的话，前 $N-2$ 条边长随机的话满足条件的概率不小。于是直接随机边长后暴力判断，重复做一定次数就行了。	时间复杂度：$O(N\lg N)$ 空间复杂度：$O(N)$
089	Codeforces 294D	Shaass and Painter Robot	给一个 $N\times M$ 的网格，给定边界上的一个格子为起点，并给出初始方向。方向只会是四个斜的方向之一。机器人会将走到的地方全部染成黑色。碰到边界会遵循光线反射的规则改变方向，问走多少步之后网格变成黑白相间的，或者指出永远不能完成。	比较容易证明网格变成黑白相间的当且仅当所有边界上应该涂黑的格子都已经被涂黑。于是在模拟的过程中记录那些边界被访问过，再记录下方向，如果在完成之前形成了环便永远不可能完成。	时间复杂度：$O((N+M)\lg(N+M))$ 空间复杂度：$O(N+M)$
090	Codeforces 240F	Torcoder	给一个长度为 N 的字符串，所有字符均为小写英文字母。现在要按顺序依次执行 M 个操作，每个操作指定一段区间，将其中的字符重新排列成字典序最小的回文串（如果不能排列成回文串忽略此操作）。求最终的字符串。	用线段树维护每段内每种字符的个数。然后对于每次操作，我们需要将区间分成两半，分别求出两边各个字符的个数然后在线段树上打懒惰标记就行了。	时间复杂度：$O(N+M\lg N)$ 空间复杂度：$O(N)$
091	Codeforces 261D	Maxim and Increasing Subsequence	给一个长度为 N 的正整数序列，其中每个数都在 $[1,B]$ 范围内。现在让这个序列重复 t 遍，求其最长上升子序列。$NB\le 2\times 10^7$。	显然答案不会超过 B。我们从小到大依次考虑序列中的数。由于以后要考虑的数都更大，此前考虑的数的值并不重要，重要的是位置，所以我们只需用 $f[i]$ 记录当前长度为 i 的子序列的最小结尾位置就行了。	时间复杂度：$O(NB)$ 空间复杂度：$O(B)$

092	Codeforces 288E	Polo the Penguin and Lucky Numbers	定义幸运数为只含 4 和 7 的正整数。现在给定两个长度相同的幸运数 $l \le r$，要求出 $[l,r]$ 中的所有幸运数按照从小到大的顺序排列后相邻数的积的和。	首先将既有上界又有下界的问题转化成只有上界的问题。之后，我们用 $res[i][j], cnt[i][j], sum[i][j]$ 分别表示最低的 i 位仅由 4 和 7 组成的方案数，并且如果 j 为 0，不能超上界，如果 j 为 1，可以忽略上界。然后考虑在一组连续的长度相同的幸运数前面加上一个数字产生的变化，就能完成转移。	时间复杂度：$O(N)$ 空间复杂度：$O(N)$
093	Codeforces 316E3	Summer Homework	给定长度为 N 的整数序列，需要支持三种操作：询问区间内每一项乘以对应的斐波那契数的积的和，单点修改，将区间内的数全部加上一个值。	斐波那契数的每一项可以分成第 0 项和第 1 项两个部分（其实可以看成是矩阵变换中间的一个状态，这个可以预处理。然后我们只需要维护每个区间内这两部分的贡献就行了。	时间复杂度：$O(M \lg N)$ 空间复杂度：$O(N)$
094	Codeforces 273D	Dima and Figure	给定 $N \times M$ 的网格，要将某些格子涂黑。要求黑格相互连通，并且任意两个黑格间的最短路长度等于曼哈顿距离。这里连通指的是四连通。	容易想到如果每一行每一列单独提出来，其中的黑格必须连通。如果从上到下依次填每一行，那么相邻行必须有相接触的部分，并且左右端点必须是先增后减。用 $f[i][l][r][k]$ 表示前 i 行考虑后第 i 行左右端点分别为 l,r，k 记录两个端点是否已经开始减小。直接转移的话是会超时的，不过可以用类似差分的方法将转移优化为 $O(1)$。	时间复杂度：$O(NM^2)$ 空间复杂度：$O(NM^2)$
095	Codeforces 303E	Random Ranking	有 N 个人参加了一场考试，给定每个人可能得分的范围，假定其得分为这个范围内等概率随机的一个实数，求出每个人在各个排名上的概率。$N \le 80$。	这道题不错。首先，考虑 N 个区间可把数轴分成一些小的区间，使得原来的区间要么和小区间不相交，要么包含整个小区间，并且任何原来的区间都可以分成若干个小区间。我们可以枚举当前要计算的人的分数在哪一个小区间内出现，然后，注意到其他的人的分数有三种情况：落在小区间左侧、中间、右侧。我们直接用 $O(N^2)$ 计入状态。左侧和右侧大小关系已经确定了，中间的也好算了，因为所有数的可能范围都一样，所以排在任何位置的概率都是 $1/M$。	时间复杂度：$O(N^5)$ 空间复杂度：$O(N^2)$
096	Codeforces 235D	Graph Game	现在执行如下操作，对于一个连通块，将其点数加入 ans，然后随机其中一个点，将其删除，重复直到所有点都被删掉。给一张 N 个点 N 条边的无向连通图，求对其操作后 ans 的期望值。	好题。一开始就想偏了，想成求每个连通块形成的概率，然后就陷死了。正确做法是注意到如果点 A 在被删的时候和 B 连通，就会对 ans 产生 1 的贡献。所以只需求出没对点出现这种情况的概率。树的情况就比较简单了，只需要 A 在 A 到 B 这条链上的点中是第一个被删的就行了。章鱼的情况需要将路径上的点分成两类：树上的（一个都不能删），环上的两侧（可以有至多一侧有删掉的点）。	时间复杂度：$O(N^2)$ 空间复杂度：$O(N)$
097	Codeforces 311E	Biologist	有 N 条狗，每条狗有初始性别。现在可以选择改变某些狗的性别，改变有代价。如果满足指定的一些狗都是某一性别的条件，便可以得到一定收益。这样的条件有 M 个。问最大收益或最小代价。	转化成最小割模型，即先假定能够获得所有收益，然后求出最小的损失。对每个条件需要新建一个点。	时间复杂度：$O((N+M)^2 M)$ 空间复杂度：$O(N+M)$

098	Codeforces 303D	Rotatable Number	求小于 X 在满足存在长度为 N 的可旋转数条件下的最大进制。可旋转数定义如下：长度为 N 的正整数，其通过旋转得到的 N 个数是自己的 1 倍，2 倍，一直到 N 倍。	这题中说的可旋转数其实就是 Cyclic Number，详见维基百科。判定其是否存在大致就是判断原根。	时间复杂度：$O(X\lg^2 N)$ 空间复杂度：$O(1)$
099	Codeforces 261E	Maxim and Calculator	一个计算器有两个整数单元 a,b。每次操作有两种选择： 1. a 赋值为 $a\cdot b$ 2. b 赋值为 $b+1$ a 初始为 1，b 初始为 0。 求 l,r 中有多少个 a 可以通过不超过 p 步得到。 $p\le 100$	由于 $p\le 100$ 所以 a 只能是不超过 100 的素数的乘积，这个数量是比较有限的。所以先将之暴搜出来。然后，我们用 $f[a][b]$ 表示某一状态的最小步数，转移时还要知道 ab 对应的位置，注意到 b 不变，ab 会随着 a 增大而增大，利用此单调性即可。	时间复杂度：运算次数上界大致为 3×10^8 空间复杂度：上界为 10^6 数量级
100	Codeforces 319D	Have You Ever Heard About the Word?	给定一个长度为 N 的字符串，不断找到其最短的形如 XX 的子串（如有多个取最靠左的），去掉其一半。求最终的字符串。	可以证明，每次迭代后最短的形如 XX 的子串的长度不会变短。因而，我们可以从小到大枚举 X 的长度，判断是否存在满足条件的子串。判断可以使用哈希，以 L 为步长取一些点，如果相邻点向前的最长公共子串长度与向后的公共子串长度之和不小于 L，则说明存在这样的串。每次发现便暴力执行删除，并重新计算哈希值就行了。可以证明此过程中发现的不同 L 数量是 $O(\sqrt{N})$ 的。	时间复杂度：$O(N\lg^2 N+N^{1.5})$ 空间复杂度：$O(N)$