IOI2014 国家集训队作业 ACM/ICPC World Finals 试题泛做

福建省厦门第一中学 刘定峰

试题编号	名称	题目大意	算法讨论	时空复杂度
2013 A	Self Assembly	本题给定了若干种正方形方块，方块的四条边上分别有一个标识，为00，A+或者A-（A可以是任何大写字母）。如果两个正方形能够组合在一起，那么组合后两个正方形贴合的边必须为恰好是一个A+，一个A-。现在使用给定的若干种方块来进行组合，并且可以对方块进行旋转和翻转。问是否可以组合出无穷大的组合体。	如果可以组合出无穷大的结构体，相当于从一个方块出发，沿着贴合的方块走，能够走到另一个跟这一方块同种的方块。因此我们进行下面的建图，对于每个方块，我们对四个标识中的任意两个，设为A(+-),B(+-)，那么我们从与A互补的标识连一条有向边给B 。反过来也连。之后判断这个图中是否存在环。如果存在环就是可以组合出无穷大，反之不可。判断环可以用拓扑排序。	时间: O((26*2)^2+n) 空间： O((26*2)^2)
2013 B	Hey, Better Bettor	在赌场中有一个赌局，赢得赌局的概率为p%（p < 50），如果赌赢了可以获得1的收益，赌输了造成1的损失。该赌局可以重复进行多次。并且如果在某个时候你的总收益为负数，那么你可以向赌场赎回x%的损失，但总共只能赎回一次。问如果你采用最佳的策略进行赌博，期望的收益是多少。	我们用f[i]表示当前收益为i的情况下（i为负数表示损失），继续赌还可以获得的期望收益是多少。如果不继续赌，那么f[i] = 0.如果继续赌，那么i>0时,f[i] = p * (f[i + 1] + 1) + (1 - p) * (f[i - 1] - 1)。 i<0时，因为负收益可以赎回，所以输赢造成的收益或者损失为1 - x，故f[i] = p * (f[i + 1] + 1 - x) + (1 - p) * (f[i - 1] - 1 + x)。i=0也类似。 初始的时候,所有f[i] = 0。接着可以反复迭代进出所有的f。但是这样效率很低。 更好的做法是，设满足f[t]>0的最小的t为f[L],最大的t为f[R]。根据上述f[i]的计算式子，可以得到f[i+1]与f[i],f[i-1]的关系式。那么f[L - 1] = 0，f[L + 1],f[L + 2] ... f[0]都可以表示f[L]的一次函数。同理f[R-1],f[R-2] ... f[0]也都可以表	设输入精度为10^-x. 时间： O(x * 10^x) 空间 O(x * 10^x) (只是找规律,无证明)

			示为f[R]的一次函数。再根据f[0]与f[1],f[-1]的关系式，就可列出二元方程。解出f[L],f[R],从而算出f[0]。 但是我们不知道L，R是多少，所以可以从L = -1,R = 1开始迭代。每次算出f[L],f[R]之后,若f[L-1]>0,另L = L - 1,若f[R+1]>0,另R = R + 1.反复迭代直到不满足上述条件为止。	
2013 E	Harvard	给定一段程序，程序中只有循环和变量操作。变量必须存储在内存中，而内存被分为b个内存库，每个内存库可以存s个变量。访问内存库0的变量需要一条指令。访问其他内存库的变量需要先用一条指令将当前内存库设为要访问的内存库（如果相同就不用设置），之后用一条指令访问。现在需要安排每个变量存在内存中的位置，使得程序运行时访问内存的指令数目最少。	本题要最小化的是在不同内存库中切换消耗的指令。 这题使用搜索解决。由于内存库0中的变量可以直接访问，因此先搜索在内存库0中的变量。之后我们对给定的程序进行处理，如果访问内存库0的变量则只要一条指令不需考虑；访问其他变量x，我们就要求出访问该变量前上一个访问的其他变量y，这样待下一步搜索后，如果x与y的内存库不同，就要表示增加一条指令切换内存库。 由于程序循环的次数可能很多，处理的时候不能模拟。而应该使用类似树形dp的算法，先根据循环的情况建立一颗树，然后进行dfs，对于每个树上的节点u，记录访问u前上一次访问的其他变量为y的次数分别为多少。这样求出在程序中先访问y，再访问x的操作有多少（x,y均不在内存库0中） 之后再进行第二步搜索，确定每个不在内存库0的变量存在哪个内存库里面。搜索结束后根据上面求得的结果，计算操作数目。这一步有一个剪枝，就是如果存储变量最少的两个内存库存储的变量总数小于等于s，那么将两个内存库合并成一个的方案更优，就可以不要计算此种情况。此外注意搜索的顺序，不要将同一种方案重复搜索即可。	时间复杂度： O(L * V!) 空间复杂度： O(L) L为程序长度， V为程序中的变量数目

2013 F	Low Power	有n个机器，每个机器有2个芯片，每个芯片可以放k个电池。每个芯片能量是k个电池的能量的最小值。两个芯片的能量之差越小，这个机器就工作的越好。 现在有2nk个电池，已知它们的能量，我们要把它们放在n个机器上的芯片上，使得所有机器的能量之差的最大值最小。	首先将输入的所有电池的能力进行排序。之后使用二分答案加上贪心。 对于任意两个电池，假设机器1两块芯片能量分别为a1,a2（a1 <= a2），机器2为b1,b2(b1 <= b2)。则区间(a1,a2) (b1,b2) 不相交。否则假设b1 < a2 <= b2，对换a2,b1两块电池的位置，那么对于机器1，两块芯片能量分别为a1,b1，对于机器2,注意到a2 > b1,所以芯片1的能量一定不减，所以经过调整后，答案更优。 而对于任意电池，假设芯片能量分别为a1,a2(a1<=a2)，则a1,a2排序后一定相邻，否则我们找到一个电池使得能量 t 满足a1 < t < a2，根据上述证明t一定不是某个芯片的能量值，把能量为t和a2的电池对换，答案更优。 所以每个机器中两块芯片的能量都只能是排序后序列中的相邻两数。所以我们先二分答案为x。之后从小到大处理，最小的和次小的一定是某台机器的芯片能量，之后我们一次检查每两个相邻元素，若差值小于等于x，那么把他们当作某台机器上的两个芯片的能量；否则就要将前一个元素放在之前已经确定最小值的某个芯片上，如果当前已确定最小值的每个芯片都已经满了，则说明答案x无解。这样即可二分判断。	时间复杂度： O(n * log(V)) 空间复杂度： O(n)
2013 H	Матрёшка	一组俄罗斯套娃指大小分别为1,2,..M（M为任意正整数）的一组玩偶，每个玩偶都可以打开，并将更小的玩偶放在里面。现在有若干组俄罗斯套娃，但是每组套娃都被拆开分成若干个单独的玩偶。现在所有的玩偶排成一列，我们知道每个玩偶的大小。现在按照一下规则来合并玩偶，我们每次选择相邻的两组玩偶，接着将两组玩偶重新组合	由于只能合并相邻的玩偶组合，所以一个玩偶组内的所有玩偶都应该出现在原序列中的连续一段。所以这题使用dp解决，首先用dp[i][j]表示将i到j直接的所有玩偶组合成一个玩偶组的最小代价。我们枚举断点为k，这样就先把i到k，k+1到j的玩偶合并，在把合并之后的玩偶组合并。代价为 dp[i][k] + dp[k + 1][j] + 合并的代价。而根据合并的规则，可以求出不用打开玩偶的数目。这个需要先预处理i到j的最小值minv[i][j],和1到i中小于等于v的数目sum[v][i]	时间复杂度： O(n ^ 2) 空间复杂度 O(n ^ 2)

		得到一个新的玩偶组，使得新的玩偶组恰好满足每个玩偶都装在规格更大的玩偶里面。重组的代价为需要打开的玩偶数目。现在要求将这一列玩偶重新组合成若干个套娃组的最小代价是多少。	。利用预处理数组可以在O(1)时间内求出。 求出dp[i][j]之后，我们在用f[i]表示将前i个玩偶整合成若干个套娃。枚举j，表示最后一个套娃为j到i直接的玩偶组成。这就要求j到i之间的数恰好为1,2,...j - i + 1同样很好处理。这样就用dp[j][i] + f[j-1]更新f[i]即可。	
2013 J	Pollution Solution	给出一个在x轴上方的简单多边形。同时给出一个以原点为圆心的圆。求出多边形与圆的交的面积。	首先，如果多边形在圆内部，直接输出多边形面积。接着求出所有圆和多边形的交点。接着按照逆时针顺序，将交点分成两类，A类表示从圆外进入园内，B类表示从圆内出到圆外。接着将所有A类点进行极角排序，从极角最大的点出发，按照逆时针顺序走多边形的边，直到找到一个B类点，这样A到B的路径加上B到A的弧构成一个曲边多边形，求出曲边多边形的有向面积（就是A往B是逆时针为正，反之为负）。但是这个曲边多边形不一定都在多边形内部。因此从找到的B类点，出发，找到B类点逆时针方向上极角大的A类点，设这个点为X。如果X同A相同，则结束。否则从X出发，在按照刚才的方法，加上曲边多边形的有向面积。	时间复杂度： O(n ^ 2) 空间复杂度： O(n)
2010 B	Barcodes	给出一个条形码序列每个区间的宽度。条形码序列由若干个宽窄区间组成，宽区间的宽度是窄区间两倍。但是由于打印的误差，实际宽度可能会相差5%。并且给出了对条形码的解码方法。要求判断此条形码序列能否正确解码，并输出解码后的序列。	首先要根据宽度序列判断每个区间是宽区间还是窄区间。那么我们可以找出宽度最大值maxw,最小值minw。如果条形码合法，那么宽窄区间都至少有一个，所以maxw一定是宽区间，minw一定是窄区间。由于误差最大为5%,那么宽区间的宽度范围不超过 [maxw / 1.05 * 0.95,maxw]，窄区间的宽度范围不超过 [minw,minw / 0.95 * 1.05]。同时还要满足宽区间宽度为窄区间两倍。如果每个区间宽度都落在上述两个范围内，那么可以得到每个区间的宽窄。之后就变成对一个01序列按照题意进行解码。如果有区间不在范围内，那么条码错误不能解码。	时间复杂度： O(n) 空间复杂度： O(n)

2010 D	Castles	给出一个树形的地图，每个节点都是一个城堡。现在我们要派出攻打城堡，每个城堡有三个属性A_i，M_i,G_i，A_i为攻打该城堡的最小人数。M_i为攻打每个城堡的后损失的人数，G_i为为攻下城堡后需要留守在城堡内的人数。现在我们可以任选一个城堡开始攻击，之后每个我们都要从当前城堡移动到另一个城堡，如果移动到未攻击的城堡则会进行攻击，扣除损失和留守的人数，剩余的人可以继续攻击其他城堡。同时，每条边之多只能从同一方向经过每条边一次。求攻下所有城堡的最小人数。	使用树形dp，先要枚举起点。之后进行dfs，对节点u进行dfs时求出攻下u及u子树上的所有城堡需要的最少人数F_u,以及攻下u及u子树上所有城堡损失和留守的总人数K_u。计算方法如下，先求出子节点的F与K。之后合并答案，由于题目表明每条边只能在每个方向经过一次，因此一旦攻击某个子树就会攻击子树中的全部城堡。所以我们只要决定攻击的子树顺序。我们使用贪心，将所以子节点v按照 F_v - K_v 从大到小的顺序进行攻击。之后根据这样的顺序计算出最少人数。而贪心的正确性证明如下，如果存在两个相邻的子节点 F_v - K_v < F_v+1 - K_v+1，使得最后总人数为W。 那么交换v和v+1之后总人数W不增。	时间复杂度： O(n log(n)) 空间复杂度： O(n)
2010 F	Contour Mapping	给出一个n行m列三角形地图，给出每个格点的高度。假设每个三角形都是一个平面。现在我们要在地图上绘制所有高度为h的倍数的等高线，已知三角形的边长为d，求登高线的总长度。注意，对于一个高度相同的平面，登高线只能绘制的平面的边缘。	我们分两类计算。 第一类是位于三角形格子内部（不包括边界的等高线）。对于一个格子，我们将三个格点的高度从小到大记作a,b,c,之后我们分别绘制a到b之间所有h的倍数，和b到c之间所有h的倍数。但是等高线的条数很多，不能枚举每条等高线。注意到所有高度在a到b之间的等高线的长度都可以使用等高线高度x乘以一个系数k表示(使用余弦定理计算系数k)。这样就可以在O(1)的时间内对所有等高线的长度进行求和（使用等差数列）。 第二类是位于三角形的边界。这部分要避免重复计算，方法是用一个bool数组来表示每条三角形的边是否是等高线的一个部分。接下来对于每个三角形格子，如果三个格点高度相等，不能算等高线。如果只有两个格点高度相等且为h倍数，那么这两个格点之间的边就是等高线。同时还要处理位于地图边界的边，地图边界的边只要高度相等且为h倍数就是等高线。最	时间复杂度： O(n*m) 空间复杂度： O(n*m)

			后根据bool数组中多少条边有标记来计算答案。这样保证每条边只被计算一次，且同一高度平面内部的边不会计算。	
2010 J	Sharing Chocolate	给定一块大小为x*y的巧克力，现在我们可以将巧克力切成若干块。现在已知切开后每块的大小，问是否存在一个可行的切割方案	使用状态压缩dp，我们用dp[x][y][st]表示把x*y的巧克力分成对应集合st的若干个小块是否可行，对于每个状态，我们枚举切开的位置，并进行更新。 注意到可行的st很少，所以我们可以用一个链表把dp[x][y]中可行的st的存下来，这样更新的时候就只要枚举切开的位置。并对切开后的两块分别枚举可行的方案并进行合并，再去重即可得到dp[x][y]。	时间复杂度： O(x*y*3^n) 空间复杂度 O(x*2^n)
2010 K	Paperweight	给出由两个具有公共底面的四面体ABCD，ABCE组合而成的一个六面体ABCDE。现在我们把六面体放置在水平面上，定义一种放置方法为稳定的当且仅当四面体的重心向任意方向移动0.2个单位长度后四面体仍不会在上移动。我们可以任意放置四面体，求所有放置方案中，四面体上的一个给定点F到水平面距离的最大值和最小值。	首先枚举ABCDE中的任意三个点确定的平面作为放置的水平面，并判断其余的两个点要在水平面同测才能放置。之后判断放置方案是否是稳定的。由于可能有四点共面的情况，故四面体与水平面的接触面可能是三角形或者四边形。而稳定的条件是重心在接触面上的投影G’必须在三角形或者四边形内部，而且G‘到多边形各条边的距离都必须大于0.2,这样既可以判断是否稳定。并容易求出F到接触面的距离	时间复杂度： O(1) 空间复杂度： O(1)
2008 A	Air Conditioning Machinery	你需要在一个指定的长方体空间内安装空调管道。你有六条相同的空调管道，每条管道均为4个单位正方体组成，管道的形状在题目图中给出。我们给定空调管道的起点和终点，需要安装最少的空调管道连接起点终点，且任意两条管道不能相交。求空调管数目的最小值。	明显必须使用搜索。从起点开始，逐一搜索空调管连接方法。由于管道的两端不同，必须枚举那一端与上一条管道相接。同时新加的管道可以在新加的管道和上一条管道连接面所在的平面内进行90度旋转，继而得到四种旋转的情况。枚举上述2×4=8种情况，并检查有无重叠即可。	时间复杂度： O(8^6) 空间复杂度： O(20^3)

2008 D	The Hare and the Hounds	给定一个图和起点终点。每条边有个一个方向，给定一个行走的规则。	进行模拟，每次如果没有走到选择点，就直接按照规则走。如果走到选择点，进不断尝试，知道走到确认点。	时间复杂度： O(N*M) 空间复杂度： O(N + M) （假设N为点数，M为边数）
2008 F	Glenbow Museum	对于任意一个直角多边形（每个内角都是90°或270°的多边形），我们定义多边形的角序列为将多边形的内角按照逆时针顺序，90°用R表示，270°用字母O表示得到的字符串。现在角序列的长度N，问有多少角序列满足该角序列能够表示一个合法的直角多边形，而且该直角多边形中必须存在一点，使得该点能够看到多边形内的每一点。	首先根据多边形的内角和，我们知道R一定恰好比O多4。同时由于多边形内存在一点能看到边上的每一点，相当与多边形存在核，那么一定不会有两个连续的O，否则就不符合了。这样子除了4个R外，其他的R都是以OR配对的形式出现。我们讨论角序列的第一位是否是OR序列中的R，如果是，则最后一位为O，之后为C((N - 6) / 2 + 4 , 4)算出四个R的位置的方案数。如果第一位不是OR序列的R，直接C((N - 4) / 2 + 4 ,4)，即可。	时间复杂度： O(1) 空间复杂度： O(1)
2008 G	Net Loss	给出一个N次多项式函数P(X)，现在我们需要使用交点横坐标为c的两个一次函数 $y = a_1x + a_0$ (-1 <= x <= c)以及 $y = b_1x + b_0$ (c <= x <= 1)组成的分段函数Q(X)来在区间[-1,1]上近似表示P(X)，使得函数 H(X) = (P(X) - Q(X)) ^ 2 在[-1,1]的积分上最小。现在给出常数c，求出两个一次函数。	首先假设两条直线交于(c,d)，那么可以得到 $a_0=d-c*a_1, b_0=d-c*b_1$ 之后我们根据对积分的表达式进行化简和计算，得到一个只剩下未知数 a_1 b_1 d的二次式，注意到三个数的取值范围都是任意实数，因此我们可以采用配方法求最小值。就是先对 $a_1^{2,}a_1$,a_1d这三项进行配方，以消去 a_1。再使用类似方法消去 b_1，最后对剩余的d进行配方。求出去最小值的d，并计算出，a_1 a_0 b_1 b_0	时间复杂度： O(1) 空间复杂度： O(1)
2008 H	Painter	给出平面上的N个三角形，若任意两个三角形相交则输出ERROR。否则，输出从任意一个三角形往外走到	容易看出，我们要对三角形按照包含关系建立一颗树，使得每个节点的父亲都是包含该三角形的最小三角形。使用扫描线算法建树，使用一个平衡树维护扫描线上的三角形，按照三角	时间复杂度： O(N Log N) 空间复杂度：

		无穷大平面要穿过的三角形数目的最大值+1.	形与扫描线Y坐标较大的交点的Y坐标从大到小存贮。按照X坐标从小到大的顺序，依次处理在扫描线上添加三角形和删除三角形的事件。添加的时候，找出扫描线上的前一个三角形，则新添加三角形的父亲一个是前一个三角形的某个祖先（包括前一个三角形本身）。这样使用倍增法就可以快速找到这个父亲，即完成建树。同时每次添加的时候同时判断扫描线上的相邻三角形是否相交，以处理ERROR的情况。	O(N)
2008 J	The Sky is the Limit	给出N个底边在x轴之内，顶点在x轴上方的等腰三角形。问所有的N个等腰三角形所组成的图形的轮廓线在x轴上方的部分的长度是多少。	我们从整个图形的最左端开始出发，走过整条轮廓线。第一边应该是最左端斜率最大的边，之后对于当前边，枚举所有斜率比当前边大的边，如果两条边相交，且交点位于当前走过的路径右侧，则表明出现了一个拐点，我们求出所有拐点中最左的一个。那么就从当前边转到拐点对应的下一条边，继而求出整个轮廓线。	时间复杂度： O(N^3) 空间复杂度： O(N)
2012 C	Bus Tour	给出一个N个点的无向图。现在我们从启点出发，按照任意次序经过剩余N - 2个点，到达终点。之后从终点出发，经过N - 2个点后回到起点（可以重复经过一个点）。要求去的路上经过的前 $[(n-1)/2]$ 个点也要同时是回来的路上经过的前 $[(n-1)/2]$ 个点。求最小的总路程是多少。	先用Floyed预处理出任意两点的最短路。之后使用状态压缩DP，分别处理f[u][st],g[u][st]，分别表示从起点出发经过集合st中的点最后到u的最短路程和从终点出发满足同样条件的最短路程。 之后先枚举去的路上经过的前 $[(n-1)/2]$ 个点的集合为 st ，接着枚举来的路上第 $[(n-1)/2]$ 个经过的点u和第 $[(n-1)/2]+1$ 个经过的点v。这样就可以用f[u][st]+dist[u][v]+f[v][~st]表示最小距离（~st表示st的补集）枚举u，v就可以求出去的最小距离。 类似地，可以求出回来的最小距离。这样就求出先经过的点集合为st的时候的答案即可解决。	时间复杂度： $O(N^2 \cdot 2^N)$ 空间复杂度： $O(N \cdot 2^N)$

2012 E	Infiltration	给出一个N个点的有向图。对于每对点i，j，有向边<i,j>，<j,i>中恰好有一条存在。若我们选中了点u，则u和所有有向边<u,v>的终点v都被控制了。现在我们需要选出最少的点，使得每个点都被控制。	对于这道题目，明显要用搜索。首先，我们可以贪心给搜索确定一个上界，贪心方法是每次选中控制当前未控制点数目最多的点。对于图的边数是C(N,2),所以根据抽屉原理，每选中一个点，图中未控制点的数目就要减半。这样上界最多是log(N) 之后开始搜索，把所有点按照可以控制该点的点的数目从小到大进行排序。接着按照顺序检查每一个点，如果未被控制，就在可以控制该点的点的集合中选一个。这样即可搜索出最优解。 可以使用bitset来压位存储每个点是否访问的信息，加快搜索。	时间复杂度： $O(C(N, log(N)))$ 空间复杂度： $O(N)$
2012 A	Asteroid Rangers	在三维空间中给出N个小行星的初始坐标向量和移动速度向量。现在我们需要在一些小行星之间建立通讯网络，使得任意两个小行星都可以直接或通过其他小行星间接通讯。建立通讯网络的代价为两个小行星之间的欧几里德距离。由于小行星不断移动，因此总带价最小的方案是不断变化的。求出方案变化的次数。	首先预处理任意两对小行星之间距离关于时间的函数，可以表示成 $d(t) = ax^2 + bx + c$ 的形式。之后枚举任意两条边x，y，求出方程 $d_x(t) - d_y(t) = 0$ 的根（要讨论方程的次数），并建立两条边x，y大小关系变化的事件。 之后我们先按照每条边的初始长度，求出最小生成树。接着按照时间将事件排序，从先往后处理。对于每个事件，如果是边y的长度开始小于边x，那么只要满足边y不在生成树中，边x在生成树中，y加入生成树后形成的环包含x这三个条件，y就会取代x成为生成树的边。这样我们使用一个布尔数组记录每条边是否在树中，并用一个动态树（因为点数少，所以暴力维护）来维护树的形态，来判断条件3是否成立。 最后注意假设若干同时放生的事件中若有多个修改，那么只计算做一次（因为是同时发生的）	时间复杂度： $O(N^5)$ 空间复杂度： $O(N^4)$

2011 A	To Add or to Multiply	现在给定两个常数a,m,并定义两种操作，操作A为给一个数加上a，操作M为给一个数加上m。我们要求一个操作的序列，使得对于任意一个位于区间[p,q]的整数，经过操作序列中的操作后，可以变成一个位于区间[r,s]的整数。求最短的操作序列，若有多个，则求字典序最小的一个。	首先，我们发现如果m等于1，则不必进行M操作，直接处理。如果m大于1，那么M操作最多进行 $log_m s$ 次，所以我们可以枚举进行M操作的次数t。之后我们假设第i次M操作前执行了A_i次A操作。A_{t+1}表示最后一次M操作后执行的A操作次数。现在我们的任务就是求出A序列。 对于给定的数x，经过上述变换后，得到 $xm^t + \sum_{i=1}^{t+1} A_i\, m^{t+1-i}$ 因此我们发现，要求出的序列应该满足 $\sum_{i=1}^{t+1} A_i\, m^{t+1-i} \in \left[r - pm^t, s - qm^t\right]$ 注意到求和式的形式类似进制分解，因此我们只要把后面区间的上下界按照t + 1位的m进制进行分解（此时最高位上的数可以m）。之后找出第一个不一致的位，则A序列之前的部分就是前面一致的部分。后面根据贪心肯定是全部0。就解决了。	时间复杂度： $O(log^2 s)$ 空间复杂度： $O(log\ s)$
2011 H	Mining Your Own Business	给出一个N个点的无向连通图。现在需要在图中某些点建立安全出口，使得从图中去掉一个点后，剩下的任何一个点都可以走到一个安全出口。求设立安全出口数目的最小值，以及方案数目。	注意到这是一个关于连通性的题目，那么我们在求出图中所有的割点。假如原图中没有割点，那么我们只要任选两个点建立逃生出口即可。这是因为任意取掉一个点后，图保持连通，而至少还保留一个出口； 如果原图中有至少一个割点，那么从图中删除所有的割点，这样图中就剩下了若干连通块。我们对每个连通块进行宽搜，求出每个连通块连接的割点数目。如果连接的割点数目为一，那么该连通块内要任选一点作出口（因为删去割点后该连通块就孤立了）。而这样的解也是符合题意的，因为删去非割点图仍然联通。而如果删去的是割点，对于连接的割点数目为一的连通块肯定符合。而对于连接的割点数目大于一的连通块，可以沿着未删割点走到一个出口。故是合法的最优解	时间复杂度： $O(N)$ 空间复杂度： $O(N)$

2007 I	Water Tanks	现在有N个水箱，已知每个水箱的体积，其中第一个水箱与敞口与大气相通，其余水箱均密闭。相邻两个水箱之间使用管子相连，并且满足连接管的高度递增。现在我们往水箱1中缓慢注水直到水溢出水箱1为止，根据物理原理，求出所有水箱内的水的总体积。	从水箱2开始，逐一模拟灌注每个水箱的过程。处理水从水箱i - 1通过管道注入水箱i时，注意到水箱i - 1中的剩余空气与水箱i及之后的水箱的空气都是相通的，假设这些空气的气压为P（P可以大于大气压，因为灌注之前水箱时已经压缩了空气）。那么根据这个气压，我们求出将水箱i加水直到与连接管高度相平时，气体气压是否能够使水通过。如果此时水压高过气压，那么水箱i和水箱i - 1中的水面继续升高，此时水箱i - 1的气体将不再与水箱i联通，另外处理。而对于水箱i判断水面是否能够上升到下一连接管的高度，如果能继续循环处理，否则退出。 此外对于水箱1注入水箱2的情况，因为水箱1与大气相通，因此开始注水时水箱2的气体可以排除到大气中。特殊处理。	时间复杂度： $O(N)$ 空间复杂度： $O(N)$
2013 C	Surely You Congest	给出一个N个点M条边的无向图，边的权值代表通过边的时间。现在有C个人同时从各自的出发点去到点1。并且每个人都必须走出发点到点1的最短路径。假设两人在同时沿着同一方向进入某一条边，那么就会发生阻塞。现在我们可以安排每个人的路径（一定是最短路经之一），使得能够不阻塞到达的人最多，求这个人数。	首先预处理出最短路图，这样每个人都一定沿着最短路图走。其次，注意到每个人都走最短路，那么两个人如果发生阻塞，那么他们出发点到点1的最短路长度应该相等。而最短路长度不同的点出发的人不会阻塞。因此只要考虑对于所有具有相同最短路长度的出发点上的人，他们彼此路径不相交就保证不阻塞。因此对每组人求一个最大流即可求出不相交的最大路径条数。	时间复杂度： $O(M * C)$ 空间复杂度： $O(M)$
2013 I	Pirate Chest	给定池塘，假设池塘的底面是一个N*M的矩形，并且已知每个单位正方形内深度。现在池塘中充满了水，我们要将一个立方体箱子沉入水中，箱子底面的一条边边长小于a，另一条边边长小于b，且满足箱子沉入水中后，水面上升能够使得箱子表面不露出水面（水面上升后不会	假设已经知道沉入池塘的位置，则求出底面中深度最浅的一格，假设该格深度为H_{min}，底面的面积为S，那么容易列出根据这两个值方程求出箱子的最大体积V。 因此现在我们根据H从大到小枚举每个格子，并且求出满足该格是最浅格的最大底面面积S。方法就是维护一个并查集，用来求出每个格子左边和右边第一个未处理到的格子（该格子深度	时间复杂度： $O(N^3)$ 空间复杂度： $O(N^2)$

		溢出）。求满足题意的箱子的最大体积。	比当前枚举格子浅）。之后我们枚举底面的上下边界，用之前预处理的信息快速求出左右最大边界即可求出答案。	
2012 D	Fibonacci Words	定义字符串序列F[0] = 0 F[1] = 1 F[n] = F[n-1] +F[n-2] (+表示字符串连接) 求给定串p在F[n]中的出现次数。	假设用G[n]表示p在F[n]的出现次数，则G[n] = G[n-1] + G[n-2] + 跨过F[n-1]与F[n-2]的出现次数。我们发现对于同奇偶的所有F串，他们的前缀和后缀都是相同的。因此预处理出奇数和偶数位置的F串，分别能与p串的那些前缀和后缀匹配。并且在处理的时候只要考虑F[n-1]为p的某个前缀，F[n-2]为p的某个后缀的情况即可。	时间复杂度： $O(N + Len(p))$ 空间复杂度： $O(N \cdot Len(p))$
2013 D	Factors	定义f(k)，表示将正整数k写成若干质因子相乘的形式后，质因子排列顺序的方案数目。现在给定m，求出满足f(n) = m的最小的n。	首先假设整数k的每个质因子分别出现了 $q_1, q_2, \ldots, q_m$ 次，那么容易求出f(k)的公式 $f(k) = \frac{(\sum_{i=1}^{m} q_i)!}{\prod_{i=1}^{m} q_i!}$ 这样我们只要先处理阶乘的质因子分解式子。之后枚举f(n)的质因子的总个数，并搜索q序列，使用质因子分解计算f(n)。之后根据得到的q序列，贪心构造n。即将出现次数从大到小排序，依次设为2，3，5，7...	时间复杂度： N/A 空间复杂度： $O(1)$
2012 K	Stacking Plates	给定N堆盘子，已知每堆中盘子的个数H_i，和每个盘子的大小，且每堆盘子在叠放时都满足盘子大小从上往下不降。现在我们要将N堆合并作一堆，且仍满足不降的性质。我们只能进行两种操作，将一堆盘子从某个盘子处分成上下两堆，或者将两堆合并成一堆（需满足一堆的最大盘子能够叠放放在另一堆最小盘之上）。求使用操作数目的最小值。	首先，假设我们先进行分堆操作再进行合并操作。那么容易发现合并的次数的分堆次数加N - 1，故最小化分堆次数即可。 先预处理，如果一堆中有多个相同大小的盘子，应该视作一个进行处理。之后我们进行贪心，从小到大处理所有堆中的盘子，如果某个两个盘子属于同一堆，而且在盘子按照大小排序后相邻，那么就可以一起移动，减少一次分堆次数。但是注意会有来自多个堆的盘子大小相同的情况，而这些盘子因为大小相同，所以顺	时间复杂度： $O(NH\ log(n))$ 空间复杂度： $O(NH)$

			序可以调动。那我们就要安排他们的顺序，使得尽量多相邻出现。	
2012 L	Takeover Wars	给定两个序列S_a，S_b，现在两个玩家A，B轮流操作。每次可以将自己序列中的两个数合并，合并后的数为原数之和；或者选择对方序列中的一个小于自己序列某一个数的数删除。问哪个玩家有必胜策略。	使用贪心策略。先将两个序列从大到小排序。 对于先手玩家A，分第一步合并或者删除两种情况。如果第一步删除，那么对于玩家B，因为当前B序列的最大值小于A，所以必须合并最大的两数。之后对于A，如果序列A的最大值大于序列B的最大值，那么A就取胜了（因为之后B不管怎么合并的数都小于目前B的最大值）；否则A也必须合并最大两数，之后对于B进行同样的考虑。直到求出胜者。 如果第一步合并，那么按照上述类似进行讨论即可。	时间复杂度： $O(Len(S))$ 空间复杂度： $O(Len(S))$
2012 F	Minimum Cost Flow	给定一个N个点的流水网络，其中有M条连接两点的水管，而每个点上有K_i个空接口。每个节点都有一个高度，我们给源点施加适当的水压，那么水可以沿着水管流到所有高度小于等于给定水压的点。同时如果水流到了某个点而存在空接口，这会导致溢出时不允许的。我们可以用0.5的代价封堵接口，或者在存在空接口的两个节点间建立水管代价为两点的欧几里德距离。问使得源的水能给流到汇且不溢出的最小代价。	首先我们必须枚举最后施加的水压，这样网络中只有高度小于等于水压的点才可以流到。 这样网络就被分成若干联通块，我们需要连接某些边使得源汇联通。这样就变成一个最短路问题，即原先图中的水管没有费用，而新加的水管费用等于距离。 不过还要考虑堵塞空接口的代价，注意到建立一条边只能阻塞两个接口，而代价至少是1，因此一定是用堵塞法封堵更优。所以我们求最短路时，如果下一个点v与当前点u不在同一个连通块内，那么我们除了计算新加水管的费用以外，还要在额外计算封堵v所在连通块内所有接口的费用，同时在减1扣除新加水管所封堵的两个接口接口。可以使用并查集维护连通块。	时间复杂度： $O(N^3)$ 空间复杂度： $O(N^2)$
2011 F	Machine Works	现在你需要管理一个机器公司。初始时你C的初始资金，并且没有机器。有N台可供购买的机器，每台机	容易列出DP公式，假设按照D_i从小到大的顺序，那么另F_i表示D_i天卖出所有机器后最大的资金,可以得到 $F_i = \max\{ F_j - P_j + R_j + G_j(D_i - D_j)$	时间复杂度： $O(N\ log(N))$

		器有四个属性D_i P_i R_i G_i。分别表示可以够买这台机器的日期，购买机器的价格，卖出机器的返还，机器每天运作的收益。现在我们选择买入机器，并且在买入机器后的第二天就可以运作机器取得收益。同一时间最多只能有一台机器，我们可以卖掉机器买新的。问D天后的最大资金数目	, $F_j >= P_j$} 根据这个公式可以在N^2时间计算。 之后是用斜率优化来优化DP。注意到DP式子可表达为， $F_i = G_jD_i + F_j - P_j + R_j - G_jD_j$ 另 $X_j = G_j$ $Y_j = F_j - P_j + R_j - G_jD_j$ 即可表达为线性规划形式。由于这题X不满足递增，所以使用CDQ分治即可。	空间复杂度： $O(N)$
2008 B	Always an Integer	给定一个N次多项式 $f(x) = (\sum c_i x^{e_i}) / d$ 问当自变量为任意整数时，多项式值是否都是整数。	原来的式子相当于 $\sum c_i x^{e_i} \equiv 0 \ (mod\ d)$ 我们将d进行质因子分解。之后考虑 $\sum c_i x^{e_i} \equiv 0 \ (mod\ p^k)$,此时p是一个质数。不难发现，如果等式左边不是一个模意义下的零多项式，那么多项式次数至少是 $\varphi(p^k)$,因此如果次数小于 $\varphi(p^k)$ 就无需考虑了。若对于所有的质因子都满足,那么暴力验证0..2N所有的数即可。	时间复杂度： $O(N)$ 空间复杂度： $O(N)$
2009 A	A Careful Approach	现在有N架飞机需要降落在机场，每架飞机有一个降落的时间区间[Li,Ri]现在我们要在时间区间内安排每架飞机的降落时间，使得相邻两架飞机降落时间差的最小值最大。	首先，我们需要二分答案。之后我们搜索飞机降落的顺序，然后我们贪心验证，对于每架飞机，在满足与前一架飞机时间差的前提下，尽量早降落，即可验证。	时间复杂度： $O(N!\ log(N))$ 空间复杂度： $O(N)$
2008 E	Huffman Codes	给定N个字符构成的文档的哈夫曼编码，现在假设每个字符的出现的百分频率都是正整数，求可能的所有频率组合的数目。(注意两个频率组合不同当且仅当所有字符的频率构成的集合不同)	根据哈夫曼编码建立哈夫曼树。假设每个树上点的权值为这点的子树内所有字母的频率和，那么根据哈夫曼树的建立方法，可以发现，深度小的点权值大于等于深度大的点，同一深度的点中，右边的点权值大于等于左边的点权值。根据这一性质，我们可以知道树节点的权值的大小关系。 之后按照字母出现频率从小到大的顺序搜索字	时间复杂度： N/A 空间复杂度： $O(N)$

			母的频率，每次搜索一个字母频率之后，要确保树节点的从小到大的顺序没有被打破。加上剪枝即可。	
2011 J	Pyramids	有两种类型的金字塔，高度为H高金字塔每层分别为边长H,H-1,...,1的正方形，矮金字塔每层为变长H,H-2,H-4,...,的正方形。现在给定石块的数目N，求出一个建造金字塔的方案，使得每块石块都用上，且金字塔数目最少。	相当于一个背包DP问题。我们用f[i][j]表示选前i个金字塔组合出使用j个石块的方案的最小金字塔数，即可。	时间复杂度： $O(N\sqrt[3]{N})$ 空间复杂度： $O(N\sqrt[3]{N})$
2011 C	Ancient Messages	给定六种象形文字。现在给定一幅n*w个像素的图像，图像中包含若干象形文字，并象形文字可能扭曲，但保持拓扑同构，要求识别出这些文字。	注意到，文字扭曲后保持拓扑结构不变。因此我们需要计算出每个文字内部“洞”的数目，方法就是先对黑色像素的联通快进行搜索，得到每个像素属于哪个文字。之后对白色联通快进行搜索，若某个白色联通块位于一个文字内部，那么就是一个洞。而六种文字洞的数目各不相同，即可识别。	时间复杂度： $O(NW)$ 空间复杂度： $O(NW)$
2011 K	Trash Removal	给定一个N个顶点多边形，现在我们可以任意旋转这个多边形，使得多边形在X轴方向上投影的长度最小。	容易证明，最优的多边形必然有某两个顶点的连线与X轴垂直，故枚举两点连线，得出旋转角度，计算即可。	时间复杂度： $O(N^3)$ 空间复杂度： $O(N)$
2011 D	Chips Challenge	给出一个N×N的矩阵，每个位置为1,0或不确定。我们需要给不确定的位置填上0或1,使得第i行和第i列1个个数相同，且每行每列的1的个数到小于等于总的个数的a/b。求填入1的个数的最大值。	假设$P_{i,j}$为所有未定的第i行第j列的值，L_i为第i行原有1的个数，C_i为第i列原有1的个数，则根据 $\sum_{j=1}^{N} P_{i,j} + L_i - \sum_{j=1}^{N} P_{j,i} - C_i = 0$ 我们把每个等式当作一个点，而注意到每个$P_{i,j}$恰好一正一负各出现一次。而$L_i - C_i$是常数，可以作为与原点汇点的连边，因此可以使用志愿者招聘的方法建图。	时间复杂度： $O(N^5)$ 空间复杂度： $O(N^2)$

			不过，每个节点的流量上界为节点总数的a/b，而节点总数我们是不知道的。但是可以用迭代的方法，先假设所有节点都是1,根据这个总数的a/b算出1的最大值，若等于总数则表示求出答案。若小于则用此最大值当作节点总数计算上界，继续迭代。 但是还有一个问题就是如何求出1的个数的最大值，在原先建立的图中，我们对于每条代表未确定点的边，都设置 - 1的费用，求出最小费用流的费用就是答案。但是这个图有负权环，需要转化成有向无环的图才能求费用流，方法WC2013已经讲过了。	
2010 G	Islands	有N座岛屿，现在已知每座岛屿的坐标，以及编号小的岛屿在编号大的岛屿西边。现在我们要找一条从1到N再回到1的路径，使得去的路上一路向西，回来的路上一路向东，使得除1和N以外的点都在去或者回的路上恰好经过一次。同时，有两个特殊点A和B，他们必须分别在去和回来的路上经过。岛屿之间的距离为欧几里德距离，求出路径长度的最小值和方案。	我们可以认为原题是找两条从1到N的路径，使得每条路经经过的点的下标依次增大。使用DP，设f[k][i][j]表示当前处理到第k个点，两条路径的末尾分别为i和j。注意到此时i和j两个必有一个是k，假设i=k，那么如果j=k-1，那么上一个状态可以是 f[k - 1][t][j] 其中t为任意小于k - 1的自然数。如果j < k-1，那么上一个状态就是f[k-1][k-1][j]。j=k同理。同时注意状态只有N^2个，无需使用三维数组，二维即可。	时间复杂度： $O(N^2)$ 空间复杂度： $O(N^2)$
2010 C	Tracking Bio-bots	给出一个N×M的地图。地图中有W堵墙。从一个空格子出发，向右或者向上走，如果不能走到右上角的出口。那么这就是一个坏格，求坏格的数目。	首先进行离散化，把地图大小缩小到W级别。之后对新的地图从右上角开始进行BFS，每次往左或者往下走，所有没有遍历到的各自就是坏格。注意可能右上角也是墙的特殊情况。	时间复杂度： $O(W^2)$ 空间复杂度： $O(W^2)$

2012 I	A Safe Bet	给出一个R×C的网格，一束光线从第一行的左侧射入。现在有两种反射镜，形状分别为'/','\'。已知网格中所有N块反射镜的位置，现在我们想在某个格子上加入一块镜子，使得光线从最后一行的右侧射出。问这样的位置的个数和字典序最小的可行位置。	首先预处理光线射入后可以到达的镜子和到达的方向，同时预处理每个镜子从每个方向射出能否从最后一行右侧射出。之后我们考虑光线从起点到某一镜子之后，经过新添加的镜子，到达终点。那么假设当前光线向上，在第R1行添加'\'镜子，那么光线就会向右，只要求出R1行右边是否有镜子能到终点。而R1可以取这个镜子到同列的上一个镜子之间的任何值。因此维护一个线段树，区间求和。	时间复杂度： $O(N \log N)$ 空间复杂度： $O(N)$
2007 A	Consanguine Calculations	人类的血型是由ABO血型系统和Rh血型系统的组合决定的，现在已经给出两种血型系统的遗传规则。根据规则，我们想求出知道父亲，母亲，孩子，三者中两者的血型后，剩下一者可能的血型	首先枚举剩下一者的可能血型，之后进行按照遗传规则判断即可。	时间复杂度： $O(1)$ 空间复杂度： $O(1)$
2009 E	Fare and Balanced	给定一个N个点M条边的有向无环图，同时满足任意一条边都在1到N的路径上。 现在我们要对一些边加上费用，使得从1到N的任意路径长度一致，且至多经过一条修改后的边。 求出修改后的最小路径代价，以及修改方案。	另dist_max(i,j)为i到j的最长路，另dist_min(i,j)为i到j的最短路。那么假设一条从u到v的路径被修改了，那么我们有dist_max(1,u) = dist_min(1,u) and dist_max(v,n) = dist_min(v,n) 成立。否则因为任意路径上之多修改一条边，那么修改后仍存在长度不等的路径。根据这个条件可以求出那些路径需要修改。 同时容易知道最小修改后路径代价为 dist_max(1,n) 那么对于一条符合修改条件的边，修改后的新权就是 dist_max(1,n) - dist_max(1,u) - dist_max(v,n) 而对于同一路径上存在多个可修改边的情况。我们始终修改拓扑序最前的一个修改边，之后的边不修改。这样就可以用拓扑排序依次求出需要修改的边。	时间复杂度： $O(M)$ 空间复杂度： $O(M)$

2009 I	Struts and Sprints	给出N个窗户，这些窗户彼此不相交，且所有的窗户都包含于最外的那个窗户里面。现在对于每个窗户，有六根弹簧或者木棒连接窗户的水平对边，竖直对边，和上下左右四条边与直接包含这个窗户的窗户对应边。满足水平的三条边长度之和为外层窗户的宽，竖直三边之和为外层窗户之长。 现在我们每次改变最外层的窗户的大小，那么所有的弹簧会被等比例的压缩或者拉伸。而木棒保持长度不变。想求出变换后每个窗户的大小同位置。	首先，要求出每个窗户的外层窗户。容易发现窗户之间的包含关系形成一颗树，我们要进行建树。可以将所有窗户按照宽排序，从大到小处理，这样对于每个窗户，包含这个窗户的宽度最小的窗户即使外层窗户。 之后我们对于每个外层窗户大小改变的事件，按照宽度从大到小的顺序遍历整颗树。对于每个窗户，根据外层窗户的新大小和新位置，计算本窗户的新大小和新位置。注意要特判三条边都是木棒的情况，依照题意其中一条会被改成弹簧即可。	时间复杂度： $O(N^2)$ 空间复杂度： $O(N)$
2007 F	Marble Game	给出一个N×N的棋盘，上面某些格子里面有标有数字的小球，另一些格子是表有数字的洞。某些两个格子之间可能有墙隔开。 现在我们可以从某个方向提起棋盘，使得每个小球都向着相反方向运动，直到小球落入洞中，遇到障碍或者被其他小球阻挡为止。注意一旦某个球进入洞中，那么这个洞就被填满了，其他小球可以经过这个格子而不落入洞里。 求出从初始状态最少经过几步，才能使每个小球进入对应号码的洞里。	使用宽搜。每个状态存储每个小球的位置，每次枚举方向来扩展状态。扩展状态时，依次扫描每个小球，判断是否能够向指定方向移动，同时注意维护每个洞是否已经填满了。 同时维护一个哈希表来记录每个状态是否出现过，在扩展状态时进行判重即可。	时间复杂度： N/A 空间复杂度： $O(N^2)$
2007 E	Network	给出N条信息，每条信息在传送时被切割成了若干个数据包，总共是M个数据包。现在我们知道数据包到达的顺序，我们希望通过一个缓冲区调整数据包的顺序，使得每条信息的数据包连续地按照数据包内容的顺序输出。我们每次可以将当前到达的数据包直接输出，或者存在缓冲区，也可以将缓冲区的数据包	做法很显然，先枚举输出信息的顺序。之后对于当前每一个数据包，如果这个包是当前正在输出的信息的第一个未输出包，就直接输出，并依次从缓冲区中输出紧随该包的数据包。如果这条信息输出完了，就换下一条信息。如果这条信息不能直接移到输出端，就存在缓冲区，等待输出。 记得每次将包移动进入和离开缓冲区时维护缓冲	时间复杂度： $O(N!M)$ 空间复杂度： $O(N+M)$

		一道输出端。求最小的可行缓冲区大小。	区的大小，就可以求出可行的缓冲区大小	
2009 F	Deer-Proof Fence	给定N个点，现在要修建一些栅栏，将点围住。使得每个点周围距离M的范围都包含在栅栏内部。可以所有点可以处于一个或者多个栅栏之中，求出栅栏总长度的最小值。	首先，如果若干个点位于一个栅栏中，那么对这若干个点求出凸包，并且凸包向外拓展M长度得到圆角凸包就是修建栅栏的方案，长度是凸包周长加上一个半径为M的圆的周长。 之后就使用状态压缩DP，使用dp[st]表示将几何st内的点全部围住的最小长度，如果st处于一个栅栏，那么直接用上述方法计算。否则，枚举st的子集，用dp[nst] + dp[st ^ nst]来优化答案	时间复杂度： $O(3^N)$ 空间复杂度： $O(2^N)$
2007 J	Tunnels	给定N个点，M条边的无向图。现在敌人从点1出发，往点0逃跑。在逃跑的过程中，我们可以知道敌人任意时刻的位置，并且可以通过删除边的方式阻止敌人到达点0。现在假设敌人采用最优的逃跑策略，求出删除边数目的最小值。	如果先删除边再逃跑，那么删边数目应该是0到1的最小割。不过现在可以边逃跑边删边。 我们使用F[i]表示从点i出发到达点0删边数目的最小值。初始时F[i]等于i到0的最小割。而对于点i，我们可以用下列方法迭代更新F[i]的值。删除x条边，使得剩余图中的点的F的最小值为y，那么F[i] = min(F[i],x + y); 这个更新的正确性是显然的。 之后我们考虑枚举删那些边是不现实的，因此应该枚举F的最小值y，将所有F值小于y的点从图中全部删去，此时应该删除边使得i与0不连通即可。则求在删点后的图求最小割。答案为 y + 最小割。 同时我们发现假设枚举用F[j]作为y的值去更新其他F[i],更新后F[i] >= F[j] 所以我们可以用类似Dijsktra算法的方法，每次找当前最小的F[j]，从图中删去并更新其他点。	时间复杂度： $O(N^4M)$ 空间复杂度： $O(M)$
2007 C	Grand Prix	给出平面上的N个点，现在第i和第i+1号点之间连了一条边，成为赛道。而这个平面实际在一个倾斜的山坡上，赛道应该要满足每段的高度都不降。不过原	原题相当于要求旋转后的赛道满足点的x坐标不降，且都大于0。那么对于任意一段赛道，我们求出赛道的方向角，应该在[-90,90]这个区间内，而假设原来的方向角为t，那么旋转角度必须在	时间复杂度： $O(N)$

		来的赛道不一定符合，因此我们希望将赛道旋转最小的角度使得新赛道符合条件。且所有的点都在地平线之上。	[-90-t,90-t]这个区间内。而由于任意点的x坐标大于0，那么点和原点连线的方位角也要满足上面的性质。 之后我们对于若干个这样的角度区间求一个交集，最后旋转角度落在这个交集之内即可。注意应该将角度区间规范化到[-180,180]之内有利于求出交集	空间复杂度： $O(N)$
2006 I	Degrees of Separation	给出N个人之间关系的无向图。现在定义这个图的分离度为任意两个不同人之间进行联系所需要经过的最少的关系数的最大值。求出图的分离度。	用一个map将输入的名字转成数字进行建图，之后对这个图用floyd算法求出任意两点之间的最短路，即可。	时间复杂度： $O(N^3)$ 空间复杂度： $O(N^2)$
2012 B	Curvy Little Bottles	给出一个瓶子，这个瓶子是由一个N次多项式函数的图像绕X轴旋转一周得到的瓶子。这个瓶子的瓶底位于函数X=XLow处，瓶口位于函数X=XHigh处。求出瓶子的体积，并且每Inc体积在瓶子上加一条刻度，输出前8条刻度线到瓶底的距离。	设这个多项式函数为F(X)，容易发现瓶子的体积为 $\int_{xlow}^{xhigh} Pi * F(x)^2$ 这样我们先求出F(X)^2这是一个N×2次多项式，之后进行积分即可求出体积。同时我们通过二分的方式也可以求出每条刻度线的位置。	时间复杂度： $O(NLog(V))$ 空间复杂度： $O(N)$
2006 G	Pilgrimage	现在若干个人将他们的钱放在一起管理。现在有一个账本，上面记载着N条操作。操作分别为: IN x 新来了x个人，每个人都需要缴纳当前的人均钱数。 OUT x 有x个人离开，需要向他们支付当前人均钱数。 PAY x 所有人的总钱数减少了x COLLECT x 总钱数增加了x乘以人数	首先根据IN，OUT操作容易求出人数最少的时候，根据任意时刻人数大于0就知道初始人数的最小值。 此外另一个要求钱数没有小数出现，而只有在涉及IN，OUT操作时才涉及到把钱数平分的操作。而注意到COLLECT，IN，OUT三个操作都不会改变人均钱数。因此只要保证两次 IN/OUT 操作之间所有的PAY操作所扣除的钱数之和可以被当前的人数整除即可。则当前人数的必须是扣除钱数之和的某个约数，我们求出所有的约数，进而求出对应的原始人数。	时间复杂度： $O(N\sqrt{K})$ 空间复杂度： $O(N)$

		现在已知钱数总是可以平分，没有小数出现。且任何时刻人数均大于0，求所有可能的初始人数。	最后对于每次平分钱数时可行的原始人数集合求出交集，就是可行的原始人数。可以通过统计每个原始人数的出现次数实现。 最后注意初始时人均钱数可以不是整数。	
2006 E	Bit Compressor	给出一种二进制串的压缩方法，就是将而二进制串中连续的一段1（这一段前后不能再有1）替换为连续1的个数的二进制表示（当且仅当替换后位数更少才会进行替换）。现在给出长度为N的压缩后的串，以及原串的长度L和原串中1的个数。求出是否存在合法的原串，以及可行的原串是否唯一。	使用DP的方法，用F[i][c0][c1]表示将压缩后前i位对应原串中c0个0，c1个1，同时原串末尾为1的方案数目（只要存0，1，2表示存在，唯一，不唯一）。 每次扩展状态时，添加一段连续0和一段连续的1。连续0的个数就是从i开始的连续0的个数，而连续1的个数则是枚举连续1在压缩之后的长度，之后把这段从二进制转成十进制即可求出连续1的个数，即可得到新状态。注意新状态的下一个字符应该为0。 由于合法的状态数目很少，因此使用一个哈希表来存储合法状态加速。	时间复杂度： $O(N^2L)$ 空间复杂度： $O(N^2L)$
2006 D	BipartiteNumbers	定义二段数为由两段组成的数，2221111（可以表示3个2和4个1），而22222，2221222不是二段数。现在给出一个数M，求出大于M的最小的二段数满足它是M的倍数。	首先定义一段数为只有一段组成的数，比如11111记作F(1,5),2222,记作F(2,4)，那么任何一个二段数可以表示成两个一段数的和或者差,比如2221111表示成 F(2,7) - F(1,4) 而2224444 表示成 F(2,7) - F(2,4). 那么我们从小到大枚举二段数的长度L，分别预处理出F(1,L)...F(9,L)对M取模后的值，那么我枚举二段数高段的值s，低段的值t，则若t < s 表示成二段数为F(s,L) - F(s - t,mLen)，故F(s - t,mLen)与F(s,L)在模M的意义下同余， 此时为了让数最小mLen应该最大。反之t > s,F(s,L) + F(t - s,mLen) 此时mLen应该最小。用另一个数组预处理出F(x,l) = t 的最小的l和最大的l即可。	时间复杂度： $O(10^2M)$ 空间复杂度： $O(10M)$

2006 B	Remember the A La Mode!	给出N种冰淇淋和M种饼片的数量。同时给出每种冰淇淋和饼片的组合的价格，现在我们需要求出将冰淇淋和饼片组合后售出的收入的范围。注意，所有的冰淇淋和饼片都需要用上，不能有剩余。	容易发现这是一个费用流问题，从源向饼片连边，容量为饼片数量，从冰淇淋向汇连边，容量为冰淇淋数量，这些边费用都为0。之后对于饼片和冰淇淋之间连接费用为组合之后售价的边，求出最小费用最大流和最大费用最大流即可。	时间复杂度： $O((N+M)^4)$ 空间复杂度： $O(N+M)$
2006 A	Low Cost Air Travel	给出M个城市之间的N条航空线路，每条线路都按顺序依次经过若干个城市。我们如果购买了某条线路的机票，那么就需要从线路的第一个城市出发，连续经过若干城市，到达某个目的地（不一定是该段线路终点），且中间不能乘坐其他航空线路。现在给出每段航空线的机票价格，求出对于给定的旅行线路，按次序经过线路中的城市的最小旅行费用。	首先容易用floyd算法求出任意两个城市之间最小旅行费用dist[i][j]。之后，容易发现原问题这是一个分层图的最短路问题，用dp[i][j]表示走完了旅行线路中前i个城市，且最后到达j点的最小旅行费用。那么每次扩展状态的时候，可以用 dp[i][j] + dist[j][k] 来优化dp[i][k] 这个状态。也可以找从j出发的航空线，并枚举乘坐的目的地k，求出中间经过了旅行线的城市的个数t，更新dp[i+t][k]	时间复杂度： $O(M^3)$ 空间复杂度： $O(M^2)$
2009 J	Subway Timing	给定一个含有N个地铁站的树形地铁网络，同时给出了树上相邻两个地铁站之间的地铁运行的耗时（单位为秒）。现在我们需要将相邻车站之间的运行时间化为分钟，其中一部分选择上取整，剩余的选择下取整。使得对于任意两个地铁站，使用化为分钟后的耗时计算出的两个地铁站的车程与实际时间（为取整）的误差的最大值最小，求出这个误差最小值的最小值。	容易想到应该二分答案，设答案为Lm，则任意两车站的误差都小于等于Lm。对于一条权为v的边，如果上取整那么赋予新的权值 60 - v mod 60，若下取整则赋予权值 v mod 60。那么两点之间的误差就是两点路径的新权值之和的绝对值。 也就是说，假设对于某个节点u，对于所有以u为LCA的路径，误差最大值，要么是某两颗子树上到根距离的最大值，要么是某两颗子树上到根距离的最小值来构成的。 接着使用树形DP来维护最大最小值。f[u][maxl]表示当u的子树上所有节点到u的新距离的小于或等于maxl时，子树上所有节点到u的新距离的最小值的最大值。对于某个节点u，我们先递归处理子树上的情况，之后用类似背包DP的方法合并。 具体来说对某个儿子v，枚举新状态的最大值为i，枚举v的maxl为j，之后在分上取整和下取整两种	时间复杂度： $O(N^3)$ 空间复杂度： $O(N^2)$

			情况得到新边权c，对于每种情况，为了保证最大值相加不超过Lm，之前的最大值不超过Lm - j - c，故之前最小值的最大值为 F[u][min(Lm - j - c,i)]，据此最小值即可判断最小值相加的绝对值是否超过Lm。还要保证Lm + c 小于等于i ，则方案就合法。求出新的最小值来更新F[u][i]。 这样就可以在二分答案后进行判断进而解决。	
2005 J	Zones	电讯公司计划建立N个通信塔，每个N通信塔的范围内能够覆盖的用户数已知。同时，有部分用户能够被多个通信塔覆盖，现在总共有M个公共服务区，每个公共服务区都被给定的若干个通信塔覆盖，且知道公共服务区内的用户数目。现在我们只能实际修建T个通信塔，求出修建方案，使得能够实际覆盖的用户数目最多。	首先，用每个通信塔的覆盖人数减去包含该塔的公共服务区人数求出只能通过该塔能够通信的人数。 之后枚举最后的方案，对于只能通过一个通信塔通信的人数直接求和。对于公共服务区，若覆盖服务区的通信塔有一个被选，那么也就上。求出可覆盖用户数即可。	时间复杂度： $O(2^N M)$ 空间复杂度： $O(N + M)$
2005 H	The Great Wall Game	在N×N的棋盘中，放置有N颗棋子。现在我们每次移动都可以将棋子移动到上下左右四个相邻的格子中，且同一格子中至多只能有一个棋子。我们要用最小的移动次数，使得N颗棋子处于同一行，同一列，或者同一对角线上。	分别枚举最后形成的是同一行，同一列，还是同一对角线。注意到，这三种情况的都是N颗棋子移动到N个位置上，因此，我们建立二分图，将每个棋子和位置进行连边，权值为移动的代价。之后使用费用流对二分图求出最优匹配即是答案。	时间复杂度： $O(N^3)$ 空间复杂度： $O(N^2)$
2005 I	Workshops	给出N个会议和M个房间，每场会议有人数和持续时间两个属性。而每个房间也有最大人数和可用的时间两个属性。现在我们现在为每个会议安排一个房间（一个房间只有一个会议）使得尽可能多的会议能安排到房间，在此基础上，最大化安排房间的会议的总人数。	进行贪心，按照房间可用时间从小到大的顺序，对于每个房间找到所有能在这个房间中的会议人数最多的一个，为这个会议安排当前的房间即可。	时间复杂度： $O(N)$ 空间复杂度： $O(N)$

2005 E	Lots of Sunlight	给定N栋楼房，自东向西排列，每栋楼房有Mi层公寓，相邻楼房的距离为Mi。同时给出所有公寓的宽W，高H。。现在给出早晨太阳升起和晚上太阳落下的时间，并假设太阳以恒定角速度通过天空，且太阳光近似认为是平行光束。现在给出Q个门牌号码，判断门牌号码对应的公寓是否存在，若存在，给出该公寓东侧或者西侧整面墙被照射的持续时间范围。	判断门牌号码容易忽略，要判断楼房编号存在，且楼层编号也存在。 之后求出照射时间，只要枚举所有东侧的楼房，若楼房比该公寓高，那么计算光线穿过楼房恰好照射的公寓东面的墙底部的时间，求出最大值。类似的，枚举所有西侧的楼房，求出最小值。中间便是照射时间的范围。	时间复杂度： $O(NQ)$ 空间复杂度： $O(N)$
2005 B	Simplified GSM Network	给出B个通信基站和C个城市的位置，以及R条位于城市之间的道路。道路视作连接两个城市的线段。每个通信基站为手机提供通信服务，而手机需要与最近的通信基站通信。这样每个通信基站都有自己的服务范围。 现在给定一个起点和终点，需要找到一条路径，使得在路径上移动时，在不同基站直接进行切换的次数最少。	这题的核心问题是确定每条连接两个城市之间的道路切换服务区的次数。假设路径连接城市a,b，那么我们先找到距离a最近的基站。接着我们每次枚举来求出沿着边从a到b之后下一个切换的基站，注意到基站的服务范围应该以两个基站之间的垂直平分线为分界，因此枚举下一个基站后，求出分界线与a到b的线段的交点，就表示切换发生点，而切换点需要在线段之内，同时要比当前基站更靠近b。若有多个要切换的基站，就找出离a最近一个切换点对应的基站。 这样求出切换次数，就对应边的权值。而题目保证点上不会发生切换。因此对这个图求出起点到终点的最短路即可。	时间复杂度： $O(RB^2)$ 空间复杂度： $O(R)$
2005 D	cNteSahruPfefrle	给出一种洗牌方法，就是将52张扑克牌分成上下两叠各26张，之后将两叠牌交错的放置。假设原来的牌分别为0,1,2,3,...,51，那么洗牌一次后会是26,0,27,1...,51,25。现在洗牌过程可能发生错误，每次洗牌时，交换某两张相邻的牌，不过错误在每次洗牌时之多发生一次。 现在给出进行过1至10次洗牌后的结果，想求出洗过几次牌，以及每次发生错	首先，容易求出在不发生错误的情况下，洗1至10次牌之后的结果。接着我们发现，假如在洗了若干次牌之后交换某两个位置上的牌，相当交换这个张牌在原序列0,1,2,.....,51中对应位置上的牌。 这样我们进行搜索，首先根据给出的结果，还原出原序列交换后的结果。之后按照从后往前的顺序搜索，每次枚举交换的是那个相邻位置的牌，并找到原序列的对应位置进行交换。到最后使得序列变成0,1,2,.....,51。	时间复杂度： N/A 空间复杂度： $O(52)$

		误是在哪次洗牌的哪个位置。要求错误次数最少，且错误发生位置构成的序列字典序最小。	中间可以进行一个剪枝，就是对于某个序列，很容易计算出交换任意两张牌使得其变成0,1,2,.....,51，用这个作为交换次数的下界。	
2004 I	Suspense!	给出两栋楼房，现在要在两栋距离为d的楼房之间架设一座吊桥。吊桥的绳索从第一栋房子的T层连到第二栋房子的J层，且绳索最低点恰好高于吊桥的桥面1米。根据物理定律，吊桥的绳索一定呈现抛物线形。 现在我们需要确定桥面的高度，使得桥面离地面至少1米，在绳索的连接点下方至少2米。同时楼房每层可能养猫或者鸟，猫可以向上跳小于0.5秒，向下跳小于3米。桥面高度同时要避免猫抓到鸟。 求桥面高度最低时的绳子长度。	对于每只位于高度H的猫，若桥面位于区间(H-3,H+0.5)时猫可以到达桥面，而对于位于高度H的鸟，桥面高度位于区间(H-0.5,H+3)时可以被桥面上的猫达到。因此桥面高度不能位于上述两类区间的并集之中。就容易求出桥面的最小高度。 假设次高度为H，那么我们先根据抛物线上两点加上最小值计算出二次函数的解析式。y = a * (x - b)^2 + c 现在我们要求出在x = 0, x = d 之间的函数图像的曲线长度。应该使用曲线积分。即用微元法 dy = 2a(x - b) dx 那么 $dLen = \sqrt{(dy + dx)^2} = \sqrt{4a^2(x-b)^2 + 1}dx$ 之后只需求解定积分$\int_0^d \sqrt{4a^2(x-b)^2 + 1}dx$ 使用换元积分法令 x = tan(u) / 2a + b 之后转化为三角函数的积分，根据公式求出原函数即可O(1)解决。	时间复杂度： $O(T+J)$ 空间复杂度： $O(T+J)$
2011 I	Mummy Madness	给出xOy平面上的N个木乃伊，假设你的起点在原点。每个时刻，你先行动，你可以移动到相邻的八个格子或者不动;之后木乃伊会移动到相邻的八个格子，使得与你的欧几里德距离最小。一旦某个时刻木乃伊与你的位置重合，你就被抓住了。现在想问采用最优移动策略的话，被抓住之前经过的最长时间。	首先容易发现一个性质，在每个时刻的末尾，你与每个木乃伊在x坐标和y坐标的差的绝对值是保持不增的，且在一直减少到0之后就始终为0。根据这个性质，我们发现，最后能否抓到与中间测率无关。 假设现在经过了t个时刻，你的位置是(x0,y0)，那么此时如果某个木乃伊i能够抓到你，当且仅当 $\left\|x_i - x_0\right\| \le t$ and $\left\|y_i - y_0\right\| \le t$ 。 因此我们只需要二分答案t，根据上述方法查找是	时间复杂度： $O(Nlog^2N)$ 空间复杂度： $O(N)$

			否存在一个能在t秒内走到的点(x0,y0)使得你不会被抓到。具体方法就是先将所有点按照x轴排序，之后某个x0，只有x坐标位于区间 [x0-t,x0+t]中的点才有机会抓到，这样我们可以考虑先处理x坐标。之后对于有机会抓到的点，假设某个点y坐标为yi，那么答案y0不能在区间[yi - t,yi + t]中。因此用线段树维护所有不可行的区间，之后在i线段树上查找是否有可行点即可。	
2005 A	Eyeball Benders	给出两幅图。两幅图分别由N条和M条平行于坐标轴的线段组成，而且线段之间不会发生重叠，只可能相交。现在我们需要判断，第一幅图是不是将第二幅图截取某个矩形后的放大一个比例形成的。题目保证对于第一幅图，至少存在一个由不同方向的线段形成的交点。	首先，我们可以任意找出第一幅图的一个交点P，并且找出对应该交点的两条线段。之后我们枚举这两条线段在第二幅图的对应线段，就可以找出这个交点P在原图中的对应点Q。 如果原图中只有这两条线段，那么直接判断即可。否则我们需要再找出第三条线段，枚举其在第二幅图中的对应线段。这个我们根据第三条线段的位置，结合之前的两条线段的信息就可以求出放大比例。这样我们只要求出第一幅图上下左右四边的边界，就可以根据点P,Q和比例r求出第二幅图中的被放大的矩形。 之后对于每条第二幅图中的线段，求出在放大矩形内部的部分，进行放大变换。判断是否等于第一幅图即可。 判断相等要注意浮点运算的误差问题。	时间复杂度： $O(N^4M)$ 空间复杂度： $O(N + M)$
2004 H	Tree-Lined Streets	给出N条街道,已知每个十字路口只有两条街道相交,且街道的端点至多落在一条街道上形成T字型陆路口.现在我们需要在街道上种树,使得相邻两颗树的距离至少为50米.且为了安全考虑,每科树距离十字路口的距离至少为25米.求出最多可以种多少颗树.	我们逐一对于每条街道计算可以种树的数目的最大值.对于每条街道,与其他所有接到求出交点,继而求出所有的十字路口,之后这就将街道分成了一段一段. 对于每一段,若两端都是十字路口,那么就需要两边空出25米之后计算种树数目最大值.若一端是十字路口,那么只要一边空出25米.最后有一种特殊情况,就是假设路上没有十字路口,那么就无需空出.这样	时间复杂度： $O(N^2logN)$ 空间复杂度： $O(N)$

			就可以计算出来了.	
2004 E	Intersecting Dates	现在为了研究股票市场，我们需要从服务器上下载一些信息．现在已经下载了NX段信息，每段信息都是记录连续的一段时间内每天的股价．现在我们还需要NR段信息，每段也是连续一段时间内每天的股价．现在我们需要求出，除去已经下载的信息，还要从服务器上下载那些天的股价信息，并将连续天数的一段时间内的需求合并在一起输出．	因为所以的日期介于1700年到2100年,所以我们可以先用一个数组处理出1/1/1700起经过每一天的日期。之后对于一个输入的日期，可以在上述数组中二分查找得出起是1/1/1700后的第几天。 对于每个输出日期都进行这样的转化，之后使用两个数组记录每天的信息是否已经取回，和是否需要。最后利用这两个数组求出还需要取回的并输出。	时间复杂度： $O(N * T)$ 空间复杂度： $O(T)$
2013 K	Up a Tree	某个人写了一段可能有错误的前序遍历，中序遍历和后序遍历的代码。在三个遍历的子过程里面，输出语句的位置正确，即前序遍历一定是先输出根，在递归调用左右子树。但是递归的部分可能存在错误，对于3个过程中的6处递归调用，每处调用都可以是前序、中序、后序中的任一一个。 现在给出使用这三个过程求出某个树的前序、中序和后序遍历。你需要求出所有可能的错误情况，和对于该情况下的正确的前序、中序、后序遍历，有多种方案时，要求字典序最小。	第一步就是要枚举错误的情况，只有6! / (2!)^3 = 90 种。之后对于每种错误的情况，我们使用一个记忆化搜索来求出是否可行即方案。 用dp(len,p1,p2,p3) 表示当前处理的子树的大小为len，p1,p2,p3,分别表示当前树的前序、后序和中序遍历的开始位置(假设原来的三个遍历都存在同一个一维数组里，若当前树的某个遍历未知那么就令它等于-1)。那么如果p1不为-1,[p1,p1+len-1]对应的就是当前树的前序遍历，依次类推。 现在我们要做的就是枚举左子树的大小s，这样即可求出右子树的大小。那么每个遍历都可以分成左右两段，如前序遍历分成[p1 + 1,p1 + 1 + s) [p1 + 1 + s,p1 + len)。 对于这两段，根据之前枚举的错误，就知道这分别是左右子树的哪种遍历。 同时注意在上一个步骤可能有两个子段对应子树的同一个遍历，那么他们必须相等。之后还要判断对于每个子树的所有遍历，他们包含的节点集合是相等的。如果同时出现前序和后序，还有判断他们头尾的字符相等（因为都是代表根）。 进行这些判断后，就可以去除明显不合法的状态，继续递归即可。	时间复杂度： $O(N^4)$ 空间复杂度： $O(N^3)$

2011 E	Coffee Central	在一个W×C的网格内部，给出N个咖啡店的位置。现在对于Q个询问，每个询问给出一个参数M，现在要求出一个点，使得到这个点曼哈顿距离小于等于M的咖啡厅的数目最大。求出这个数目，和这个点。（若有多个，求出y坐标最小的，在此基础上，求x坐标最小的）	对于某个咖啡厅x0,y0，所有与他曼哈顿距离小于等于M的点恰好都在一个斜45度的正方形内部，如果我们给每个点新坐标 (x0 + y0,x0 - y0),即用对角线的标号来表示点的坐标，那么所有曼哈顿距离小于等于M的点就以(x0 + y0,x0 - y0)为中心，边长为2M的平行于坐标轴的正方形中。 此时我们相当于要对一个正方形中的所有数加1,如果我们进行差分，那么就相当于对四个边界点分别+1或者-1。之后处理玩所有咖啡厅之后，对差分数组进行求和，求出每个点的咖啡厅数目即可。	时间复杂度： $O(NW^2)$ 空间复杂度： $O(W^2)$
2009 D	Conduit Packing	现在给出四条圆柱形缆线的半径，现在我们需要把这四条管线装进一个大的圆柱形管子内部，求出能够装下这四条缆线的圆柱形管子的最小半径值。	首先，容易想到的是先二分答案，之后就需要判断管子半径为R的时候是否可行。 我们可以假设每个小圆（缆线）是逐个放置进入大圆（管子）内部的，那么我们先枚举4的全排列种放置顺序。之后，我们认为每个新加入的小圆与其他两个大院或者小圆相切，若是第一个加入的小圆则只与大圆相切。这样我们对于每个小圆，我们可以枚举所有的可行位置，最后判断小圆是否两两不相交，且小圆都在大圆内部即可。	时间复杂度： $O(1)$ 空间复杂度： $O(1)$
2004 G	Navigation	现在我们要对一个物体进行定位，已知N个定位信号的信号源在0时刻的位置和移动方向，且每个信号源的速度都是100。同时给出物体在T时刻时从每个信号源处接收的信号是在Ti时刻从信号源发出，且信号的速度为350。若能定位，求出物体相对于目标点的方向。否则输出信号存在矛盾或者信号的信息不足以确定位置。注意，由于时钟同步的误差，你应该把两个相距0.1的点视为一个点。	先求出每个信号源在发出信号时的位置，之后就相当于求出N个圆的圆周的公共点。 我们先找出两个不重合的圆，若这两个圆外离或者内含那么显然无解。否则我们就要找出这两个圆的公共点，如相交就是求出交点，相切时就是切点。而最终答案是所有圆的公共点，所以也是这两个圆的公共点，因此只要对上面求出的公共点进行检验，看是否在每个圆圆周上即可。 不过由于题目中给出一个0.1的误差，因此我们在判断圆与圆的位置关系的时候进行特殊判断。如两个距离小于0.1的外离的圆要当作外切处理，两个距离小于0.1的内含的圆要当作内切处理，距离	时间复杂度： $O(N)$ 空间复杂度： $O(N)$

			圆心距离与半径相差小于0.1的点都要当作在圆上。	
2003 J	Toll	给出N个城市和村庄，M条双向道路连接城市和村庄。现在你需要从起点运送一匹货物到终点，在运送的过程中，每进入一座村庄，就要缴纳一件货物作为税款。而进入一座城市，就要每20件货物要缴纳1件作为税款（不足20件的部分也按照20件计算，如21件要交2件） 已知目的地需要的货物数目，求出在起点出发沿着最优路径行走，至少要到几件货物。	容易证明，虽然携带不同数量的货物的时候最优路径不一定相同，但是只要初始货物数大于最少初始货物数都是可行的。因此满足二分性质 首先二分答案，之后就变成在一个图里求最长路的问题，用dist[u]表示每个点缴税后的最大货物数目，用Dijkstra算法求出到终点的最长路即可。	时间复杂度： $O(N^2)$ 空间复杂度： $O(N^2)$
2003 H	A Spy in the Metro	某城市的地铁网络只有一条双向地铁线路，地铁线路中有N个车站，现在已知相邻两站之间地铁的运行时间，以及每班列车的开出时间表。现在一个间谍从车站1出发，在T时刻到达车站N与另一间谍接头，期间间谍可以任意正向或者反向地乘坐地铁，使得间谍在月台上等待时间最短，求出最短的在月台上的时间。忽略地铁停站和换乘的时间。	假设我们根据时间T建出分层图，那么直接在分层图上求出最短路即可。 具体而言，用dp[t][u]表示在时刻t在车站u月台上的最小月台时间。若上一个时刻也在u，那么用dp[t-1][u] + 1更新答案;若t时刻有列车经过本站，那么假设是正向列车，那么用dp[t - dist(u-1,u)][u-1]更新答案，反向列车也是类似。 为了快速判断是否有列车经过，可先预处理一个数组记录每个车站每个时刻的列车情况即可。	时间复杂度： $O(NT)$ 空间复杂度： $O(NT)$
2003 F	Combining Images	给出一种叫做四分树的压缩图像的方法，即对于一个边长是二的幂次的黑白图像（0表示白1,表示黑）。若图像只有一种颜色，那么输出”0” + 颜色。否则输出1,接着把图像分成左上象限、右上象限、左下象限、右下象限，依次输出他们的四分树编码。现在给定长度不超过L的两幅图的四分树编码，求出他们并的编码。	根据编码，容易建立出四分树。 之后合并两颗四分树。若两颗树的根节点都是颜色不一的情况，那么就分别合并四个象限对应的子树。如果至少一个颜色一致，那么假设颜色是黑色，最后合并的结果就取决于另一颗树，直接返回另一颗树。若是白色，则合并后就是全白，直接返回。 如此递归合并，注意合并后原先颜色不一的节点可能变成颜色一致（全白），因此要特殊处理。	时间复杂度： $O(L)$ 空间复杂度： $O(L)$

			之后输出即可。	
2004 D	Insecure in Prague	给出一个加密长度n的串的方法. 首先，发送方需要选择一个整数m，满足m≥2n；以及4个整数s, t, i, j，满足0≤s, t, i, j < m且i<j。之编码的方式是这样的: m作为密文c的长度，一开始c包含m个空位(标记这些空位为0~m-1). 接下来，p的第1个字符将被放置在c的第s个空位上，而第k(k≥2)个字符将被放置在c中第k-1个字符后连续跳i个空位后的第1个空位上. 如果跳到了c的末尾，则转至c的开头处.接着我们重复上述操作再次将原串串填入密文中，不过参数s,i被替换为t,j。 现在给出一段密文，我们需要求出这段密文对应的长度最长的明文。	这题主要考察枚举。首先从大到小枚举答案串长度n，之后枚举参数s,i,求出第一次输入的明文，之后枚举参数t,j求出第二次输入的明文，并判断是否相等。 不过我们注意到，如果参数s,i,t,j后，如果我们暴力找空位来求出明文的话是非常耗时的。因此我们应该先预处理一个数组 pos[len][d][k] 表示当密文串长度为len，参数s=0,i=d时，明文第k个字符在密文中的位置。而对于s不为0，相当与每个位置都右移了若干位，因此 (pos[len][d][k] + s) % len 即是位置，这样就实现O(1)找出位置。 而枚举t,j求出第二次明文时也用pos数组，不过注意到此时第一次明文已经占据了一些空位。我们应该把第一次明文的空位全部删除之后得到一个新串，而后根据新串的长度 len2 去 pos 数组中查找对应的值。这样可以减少一个数量级的复杂度。	时间复杂度： $O(N^6)$ 空间复杂度： $O(N^3)$
2008 I	Password Suspects	现在有一个长度为N的小写字母组成的字符串S，给出M个字符串构成的字符串集合，满足集合中的每个串都是S的一个子串（每个子串的出现位置可以互相重叠）。求出所有可能的串S的数目，若数目小于等于42,求出所有可能的方案。	容易想到应该对M个字符串构成的集合建立一个AC自动机，在AC自动机的每个节点node上存储set[node]这个节点对应的字符串的所有后缀对应了集合M中的哪些串。 之后在AC自动机上进行DP，用f[len][node][st]表示当前处理串S的前len位，这len位在AC自动机上的对应节点为node，且目前已经出现的集合为st（状态压缩表示）。这样我们只要枚举第len+1个字符c，求出节点node加上字符c后的转移的节点node2,就可以更新下一个状态f[len + 1][node2][st | set[node2]] 最后输出方案时只要倒着操作，每次枚举上一个状态和上一次添加的字符，递归处理即可。	时间复杂度： $O(N^2M2^M)$ 空间复杂度： $O(N^2M2^M)$

2002 H	Silly Sort	给出一个长度为N的元素两两不同的序列，序列需要被排成升序。现在我们每次可以交换任意两个数，交换的代价是两数之和。求出排成升序需要的最小总代价。	首先，我们应该求出排成升序后每个数对应的是原来序列中的第几个数。这样我们就求出了将原序列变为新序列的置换。 将置换分解为若干个循环，为了使得代价最小，j进行贪心。所以我们交换某个循环的时候有两种方法。第一是取出循环内的最小元Min，将它依次与其他元素循环内进行交换，假设循环长度为L，循环内元素总和为Sum，代价为 Sum+(L-2)*Min. 第二种方法是，现将全局的最小值Gmin与循环内的最小值Min交换，之后用Gmin一次与循环内的元素交换，最后Gmin与循环内的最小值Min交换。这样代价为 Sum - Min + (L - 2) * Gmin + (Min + Gmin) * 2. 取较小的即可。	时间复杂度： $O(N \log N)$ 空间复杂度： $O(N)$
2003 D	Eurodiffusion	给出一张10×10的网格图，现在有C个国家分布在网格图中，每个国家为网格图中的一个矩形区域，包含在矩形区域内的点为这个国家的城市。任意两个国家不会有公共部分。 现在我们要处理一个货币流通的问题，每个国家都发行自己的欧元硬币。现在每个城市开始时都有一百万个本国硬币。之后每天，每个城市都要向上下左右的四个相邻城市（如果存在的话）支付一定数目的每种硬币，数目为该城市持有该种硬币的数目处以1000去下整。 经过若干天的流通，假设某个城市在某一天开始拥有所以国家发行的硬币，那么该个城市在该天“已经完成”。若某个国家所有的城市都完成了，则称该国家“已经完成”。求出所有国家的完成时间。	首先，我们可以先写一个程序看一下硬币的流通速度，发现对于一个10×10的网格图，即使在最坏情况下（即某个国家只有一个城市，且该城市位于地图边缘），硬币流通至所有城市所需要的天数只要几千天。 因此我们完全可以直接模拟每天的流通情况，这样不会超时。 为了方便处理，我们每次只考虑一种国家的硬币。初始时该国的城市都有1000000枚，其他国家城市无这种硬币。之后每天按照题意进行模拟，直至所有城市的“已经完成”即可。	时间复杂度： $O(C)$ 空间复杂度： $O(1)$

2003 B	Light Bulbs	给出N个排成一列的灯泡，现在有N个开关，第i个开关可以控制第i-1,i,i+1个灯泡。即按下这个开关，开关控制的灯泡的开关情况就会发生变化。 现在已知初始时每个灯泡的情况和结束时每个灯泡的情况，求出按下开关数最少的操作序列使得初状态变成末状态（相同情况下要求字典序最小） 注意为了压缩数据，灯泡的开关状态用0（关），1（开）表示，并将01串化为十进制数输入。	第一步是将十进制的序列转换为二进制以得到初末状态。由于输出的数字最多有一百位，在进制转换时需要使用高精度运算。 之后我们假设得到长度为L的二进制序列。首先，我们可以枚举开关1的操作情况，从而得出灯泡1,2操作后的情况。之后我们注意到此时只剩下开关2能控制灯泡1,因此开关2是否按下，取决与灯泡1此时的状态与目标状态是否一致。依次类推，我们可以逐一确定开关2,3,4,...L的操作情况。且此时灯泡1,2,3,...L-1都变成目标态，此时我们只要检验最后的灯泡L的状态正确即可。 最后同样使用高精度完成答案的操作序列二进制转化十进制。	时间复杂度： $O(L)$ 空间复杂度： $O(L)$