姓名：罗干

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 试题编号 | 名称 | 题目大意 | 算法讨论 | 时空复杂度 |
| 1998B | Flight Planning | 飞机飞行有n个航段，每个航段都已知其长度a，在20000英尺时的风速b、40000英尺时的风速c，要给每个航段确定一个飞行高度（1000的倍数），使得总燃料消耗最小。已知飞机速度为400海里每小时，最佳飞行高度为30000英尺，该高度飞行消耗燃料2000加仑每小时，在该高度以上或以下每1000英尺多消耗20加仑，飞机每升高1000英尺消耗燃料50加仑，下降不消耗，风速是线性函数，实际飞行速度为飞机速度+风速。  1<=n<=100，a<=100000，|b|、|c|<=100。 | DP  由于飞行高度都是1000的倍数且只能选择20000到40000之间的数，因此飞行高度只有21种。  先预处理出每个高度飞1小时消耗的燃料量。设f[i][j]表示已经飞行了i-1个航段，第i航段选择的飞行高度是j\*1000，根据j我们可以确定当前高度的风速以及实际飞行速度，这样也意味着确定了该航段的燃料消耗量w1。枚举第i-1个航段选择的高度k，根据j和k我们可以确定飞机升降消耗的燃料量w2，f[i][j]=min{f[i][j],f[i-1][k]+w1+w2}，其中f[1][j]特殊处理即可，同时要记录下方案。 | 时间：O(212n)  空间：O(21n) |
| 1998D | Page Selection by Keyword Matching | 有三种操作P、Q、E：P操作表示增加一个网页，其后跟着网页的关键字；Q操作表示询问，其后跟着询问的关键字；E操作表示结束。网页与询问的关键字重要程度按输入顺序均从8开始递减，网页与某询问的相关性的值是所有相等的关键字对重要程度之积的和（比较忽略大小写），对于每个询问输出相关性最高的5个（或不到）网页编号。  每个关键字长度<=20，网页与询问均至多8个关键字，网页数量m<=25，总操作数为n。 | 字符串处理+枚举  对于每个询问，分别求出当前询问与之前出现过的网页的相关性，再对相关性排序输出即可。 | 时间：O(nm)  空间：O(m) |
| 1998E | Petri Net Simulation | 有n个点，每个点都有一个权值，有m种操作，每个操作都是要求若干点权值-1，若干点权值+1（至少1个点要-1，至少1个点要+1），要保证每次操作后每个点权值均非负，有一个操作次数限制limit，问在limit内至多操作几次还是可以操作至少limit次，同时输出操作完成后正权点的权值，保证最后结果唯一。  n<=100，m<=100，limit<=1000。 | 模拟  由于最终的结果唯一，我们直接模拟即可，每次枚举某个操作，看看是否操作完会导致负权点出现，不会就直接进行此操作，直到操作limit次或没有可以进行的操作为止。 | 时间：O(nm\*limit)  空间：O(nm) |
| 1998G | Spatial Structures | 给定一个n\*n的像素图（0为白点1为黑点）或者像素图边长n和像素图对应的四分树的黑点位置，求出对应的四分树或者像素图。像素图对应的四分树是通过将图片不断地分割成四个大小相等的象限来构建的，如果一个象限内的所有像素点颜色都相同则不再分割，四分树的根代表了整个像素图，非叶子节点都有4个儿子分别对应了该节点表示象限的四个子象限，叶子节点表示其对应区域都是同种颜色。  1<=n<=64且n是2的幂。 | 分治  对于像素图转成四分树的情况，我们从最大的块开始分治：若当前块内有0和1，则继续分治处理该块的四个象限；若当前块只有1，则求出对应的四分树的点编号；若当前块只有0，则什么也不做。按字典序输出求出的点的编号即可。  对于四分树转像素图的情况，我们也从最大的块开始分治：若当前块不是四分树的黑色节点，则继续分治处理其四个象限；若当前块是四分树的黑色节点，则将该块均标记成1；若当前块不能再分了就停止分治。输出转好的像素图即可。 | 时间：O(n2logn)  空间：O(n2) |
| 1999A | Bee Breeding | 给出一个蜂巢的结构，蜂巢编号从1开始顺时针绕圈，形成一个由正六边形构成的图，问某两个编号对应的巢室之间的距离。  编号<=10000。 | 图论  设我们的图要有x个阶段，有n个点，则1+6x(x-1)/2>=10000，算出x>=59，因此我们的图有n=10267个点。而每个点至多与比它小的3个点连边，我们的图至多30801条边。  所以我们只需要按照蜂巢的结构，以一个圈为一个阶段，构造出整个图，接下来直接广搜查找最短路就可以了。 | 时间：O(n)  空间：O(n) |
| 1999C | A Dicey Problem | 给出一个n\*m的骰子地图，一个标准的正六面体骰子在其地图上移动，且只能在相邻两格间移动，地图上有些位置不能到，有些位置可以随意到，有些位置要保证从别的格子移向它时，骰子在那个格子时向上的一面要为这个格子的给定数值，给定初始骰子摆放形式和位置，求出一条从初始位置绕一圈回到起点的路径（或无解），保证答案唯一。  n,m<=10。 | 搜索  首先要处理出骰子上方为x前方为y时左边的数字，这样我们就确定了骰子的摆放方式。  设p[i][j][x][y]表示在第i行j列，骰子上方为x前方为y，这个状态是否可达，u[i][j][x][y]则记录下从什么状态转移过来。  我们从起点开始做BFS，因为路径唯一，我们不必考虑路径的优劣，只要能转移回起点就可以输出答案。要注意初始的状态也是在起点处，我们不能让初始状态对应的p数组位置在初始时变成true；而最后路径有可能构成了一个环，在输出时要注意起点位置恰好出现2次，否则会死循环。 | 时间：O(nm\*62\*4)  空间：O(nm\*62) |
| 1999D | The Fortified Forest | 有n棵树，已知每棵树坐标(xi,yi)、价值vi和高度li，现在要用栅栏围住这些树（栅栏材料也是这些树，1单位高度树对应1单位长栅栏），问要保留最大价值的树的砍伐方案以及可以剩余的栅栏长度。  1<=n<=15，所有数均是int类型。 | 搜索+计算几何  先枚举哪些树被保留下来，对于每种情况都求出剩下树的凸包的周长，若砍掉的树高和>=凸包周长则该方案可行。记录下最优的方案就行了。 | 时间：O(2nn2)  空间：O(n) |
| 1999E | Trade on Verweggistan | 有n堆货物，每堆货物从上到下依次有m个产品，每个产品都有其价格，你现在要买一些产品，你买进后再卖出的价格一定都是10，任意一件产品你若要买都必须将其上方所有产品都买进才行，问最大的利润和利润最大时需要买产品的数量中最少的10个答案。  n<=50，任意一堆的m<=20。 | 贪心+DP  对于每一堆货物，我们先贪心求出一堆最大的利润（直接从顶到底累加，然后求最大值），全部加起来便是总体最大的利润。  再对于每一堆货物，求出所有达到该堆最大利润可能的产品购买数量，再做一遍DP，我们设f[i][j]表示做完i堆选了j个产品是否可行，若当前堆是买k个则转移式为f[i][j]|=f[i-1][j-k]，最后输出最少的10个i（或者不够）满足f[n][i]=true就行了。 | 时间：O (nm2)  空间：O(nm) |
| 1999H | flooded! | 有n\*m个格子，每个格子都有一个高度且都是100平方米的面积，现在有k立方米水，保证水先淹没低处，问最终积水高度和积水格子百分比。  1<=n,m<=30。 | 模拟+数学题  先将格子按高度从小到大排序，对于相邻两个高度的区间我们看当前剩余水是否能完全填满，若能就继续否则就结束，这样就能解决本题。 | 时间：O(n2\*m2)  空间：O(n\*m) |
| 2000A | Abbott’s Revenge | 给出一个箭头迷宫，每个点都可以用二维坐标(x,y)表示，总共有NSEW四个方向，同时给出了从某个方向到达某个点接下来可以选择出去的方向，给定起点终点和出发时选择的方向，请找出一条从起点到终点的最短路径。  1<=x,y<=9。 | 搜索  设p[x][y][i]表示是否可以从i方向到达点(x,y)，根据输入我们可以知道从某个方向进入该点可以选择的出去的方向，因此可以使用BFS进行处理，当我们的BFS找到一个状态在终点位置时，输出当前的路径就行了。 | 时间：O(9\*9\*4\*4)  空间：O(9\*9\*4) |
| 2000B | According to Bartjens | 给定一串数字，要求在中间添加+-\*号（至少1个，且添加完后所有数不能有前导0），使得式子按正常优先级运算时答案是2000，按字典序输出所有符合的式子。  数字数n<=9。 | 搜索+模拟  先枚举每个数字后面添加的运算符号类型（或不加），由于+-\*与数字0-9的ASCII顺序是\*+-0123456789，因此我们枚举的顺序是\*、+、-、不加符号，这样在遇到符合条件的式子时直接输出就行了。  再对式子进行判断，去除有前导0和1个符号都没有的式子。  接着对式子结果进行计算，若结果为2000就输出当前式子。 | 时间：O(4nn2)  空间：O(n) |
| 2000C | Cutting Chains | 有n个环，告诉你一些两个环套在一起的信息，你可以将一些环解开重新安排它与其他哪些环套在一起，问最少解开环的个数使得最终可以让这n个环变成一条链。  1<=n<=15。 | 搜索  首先要去掉重边。  枚举哪些环被解开，对于每种情况，先去除剩余环之间存在环或者某个环与超过2个环套在一起的情况，接着统计出剩余环本身有的链数，若链数>解开环个数+1则该方案不可行。  输出所有可行方案中解开环个数最少的即可。 | 时间：O(2nn2)  空间：O(n2) |
| 2000E | Internet Bandwidth | 给定一个无向带权图和起点终点，求出起点到终点的最大流  点数n<=100，边数m<=n\*(n-1)/2。 | 网络流  直接构出流网络做最大流即可。 | 时间：O(n2m)  空间：O(n+m) |
| 2000F | Page Hopping | 给定一个n个点的有向图，保证任意点可以到达其他所有点，求出所有点到其他点的最短距离平均值。  1<=n<=100。 | 图论  做一遍floyd即可求出从所有点到其他点的最短距离，将这些数加在一起求平均数便是答案。 | 时间：O(n3)  空间：O(n2) |
| 2001A | Airport Configuration | 机场有一排n个出发门和一排n个到达门，编号为i的出发门和j的到达门距离为|i-j|+1，有n个城市，每个出发门和到达门都对应了一个城市，告诉你城市间的客流量以及m个门与城市对应的方案，求出每个方案的客流指数（人数\*距离之和），并将这些方案从优到劣排序。  1<=n<=25，1<=m<=20。 | 模拟  对于每个方案，我们都求出客流指数，再根据客流指数对方案排序即可。 | 时间：O(n2m+m2)  空间：O(n2+m) |
| 2001B | Say Cheese | 在一个无限空间中有n个球形空间（可以相交），在球形空间内移动不用时间，在球形空间外每走1单位花费10单位时间，给定起点和终点问从起点到终点的最少时间花费。  n<=100。 | 图论  这道题是最短路问题，我们先将起点和终点视为半径为0的球形空间，我们发现两个球形空间的最短路径为两者球心的连线，其长度为这两点距离-两个球形的半径和（出现负数即两个空间相交，路径长为0），然后求出起点到终点的最短路就行了。 | 时间：O(n2)  空间：O(n) |
| 2001F | A Major Problem | 给出原曲调和目标曲调的大调音阶，问原曲调和目标曲调大调音阶是否合法，若均合法，则回答一些音符在原曲调中的位置对应的目标曲调中的音符是什么。 | 模拟+枚举  先确定原曲调和目标曲调的所有音符，判断是否合法，若合法再找出询问音符在原曲调中的位置，求出对应的音符。 | 时间：O(1)  空间：O(1) |
| 2001H | Professor Monotonic's Network | 有n个数和m个操作，操作按先后顺序输入，每个操作有两个数ai和bi，会比较ai和bi位置的数并将较小的放在ai较大的放在bi，多个操作可以同时进行，前提是在该操作前的有ai或bi的所有操作已经完成，问这些操作能否完成对着n个数的排序工作，以及这些操作至少进行的轮数。  1<=n<=12，0<=m<=150。 | DP+模拟+枚举  首先计算至少进行的轮数：设f[i]表示到i操作至少进行的轮数，f[i]=max{f[j]+1}（j<i且i操作和j操作都对某位置进行了操作），f[i]至少为1。f[i]的最大值便是至少进行的轮数。  由于n较大，不能枚举n!次，可以随机若干个n的排列，对于这些排列都通过m个操作模拟一遍，若最后有任意一个不是排好序的就不能完成排序工作。  设随机次数为k。 | 时间：O(n2+km)  空间：O(m+n) |
| 2002A | Balloons in a Box | 已知一个长方体盒子两个对角的坐标，以及n个点，你可以在这n个点放气球，气球会不断膨胀形成以该点为球心的球直到表面碰到盒子边缘或其他气球，你选择的点不能在盒子外或在别的气球内，问气球占据总体积最大时盒子剩余的体积。  1<=n<=6，所有坐标绝对值均<=1000。 | 计算几何+枚举  首先枚举放气球的顺序，再按照该顺序依次放入气球，算出其最大半径，再将气球体积求和，找出体积和最大的，用盒子体积减掉便是答案。 | 时间：O(n!n2)  空间：O(n) |
| 2002C | Crossing the Desert | 在平面直角坐标系上有一些点，其中有一个起点一个终点，只有起点可以购买食物，而水在给出的点都可以获得，所有给出的点也可以存储食物，你走x单位距离就会消耗x单位水和x单位食物（x为实数），同时你最多携带limit单位的食物和水，问从起点到终点消耗的最少食物量。  点的个数<=20。 | 数论+图论  设点的个数是n，我们从终点出发做最短路，每个点记录下从该点出发去终点的最小食物花费，由于我们的食物花费量在状态转移时是不断递增的，我们只要每次选择未被选过的最小食物花费点转移就行了，这就是一个最短路问题。  在转移的时候，设两点间距离为len，我们可以明确的一点是我们在实际操作中一定是不断往返将食物从前一个点运送到后一个点，其中最后一次可以运送limit-len\*2的食物，其他几次只能运limit-len\*3的食物，因此我们只要先假设最后一次全部送满，就能确定之前还要送几次，对于多送的那些食物，再在最后一次中减掉就行了。由于都是实数运算，细节上的处理也有些麻烦。 | 时间：O(n2)  空间：O(n) |
| 2002E | Island Hopping | 有n个岛，它们的坐标是(xi,yi)，每个岛上都有居民si人，1号岛是主岛，现在要在岛间修路，使得路总长最短且使所有岛可以到主岛。路同时开始修且路修完的时间与两岛距离是1:1关系。问所有人可以到达主岛的平均时间。  1<=n<=50，其余数都是整数且在int范围内。 | 图论  先求出这n个岛的最小生成树，然后从1号岛做一遍dfs遍历这棵生成树，每个岛上的人可以到达主岛的时间即其到1号岛路径上的所有路的距离最大值。 | 时间：O(n2)  空间：O(n) |
| 2002H | Silly Sort | 给定n个不相同的整数ai，每次交换代价为交换两数之和，问将这n个数从小到大排序的最小代价。  1<=n<=1000，1<=ai<=1000。 | 贪心+模拟  先确定每个数应该放的位置，对于每个数指向其该放的位置，则形成了若干个环，我们分别处理每个环。有两种方法：用环上最小数交换一圈或者用整体最小数去依次交换，对于每个环用较优的策略处理即可。 | 时间：O(n2)  空间：O(n) |
| 2003B | Light Bulbs | 有n盏灯和n个开关，第i个开关控制i-1、i、i+1三盏灯（其中第1盏和第n盏不能控制第0盏和第n+1盏）。我们用0表示灯灭1表示灯亮，用0表示按过开关1表示没按过。给出初始状态和结束状态的十进制情况，求出先满足最少开关使用次数再满足十进制下值最小的方案。  状态的十进制位数不超过100。 | 递推+枚举+高精度  我们先将十进制状态转换成二进制，然后枚举第一个开关是否使用，我们便可以递推出剩下n-1个开关状态，再根据第n盏灯判断是否合法，求出所有合法的方案中最优的输出。注意n=1要特殊处理。 | 时间：O(n)  空间：O(n) |
| 2003D | Eurodiffusion | 你需要写一个程序来模拟欧元硬币在欧洲的传播过程，仅考虑单一的欧元面额。给出一个矩形网格，欧洲城市都在格点上，每个城市最多可以有4个邻接的城市（东西南北各一个）。每个城市都属于且仅属于一个国家，而且一个国家所有城市在平面上刚好组成一个矩形。一开始每个城市都只有一百万个本国硬币，每一天每个城市都按照它在这一天开始时的余额，将一定量的硬币送给它的所有邻接城市。这一定量的硬币是对于这个城市目前所拥有的每种图形的硬币，每满1000个就要拿出来一个。当某个城市中，每个国家的硬币都至少出现了一个就称这个城市已经完成。当一个国家的所有城市都已经完成的时候，就称这个国家已经完成。求出每个国家的完成时间。  1<=国家个数n<=20，1<=所有城市所在格点坐标x,y<=10。 | 模拟  对于每个国家的硬币分别处理，一开始将当前硬币对应的国家的城市均设成有1000000个，其余城市均设成0个，然后模拟每一天的流通情况，当某个国家所有城市硬币数均大于0时记录下当前天数，直到所有城市硬币均大于0为止。每个国家的完成时间是该国每种硬币的完成时间的最大值。  设每种硬币完全流通的时间为ti，T=∑ti。 | 时间：O(102T)  空间：O(102) |
| 2003F | Combining Images | 一种图像压缩算法是基于四分树编码的，进行四分树编码的图片必须是一个二进制像素方阵（每个像素的值都是0或1，代表颜色），方阵的边长必须是2的幂次。  如果一个图像中所有像素的颜色都相同，这个图像的四分树编码以1开始，然后是像素的颜色；如果一个图像中含有不同颜色的像素，这个图像的四分树编码以0开始，然后依次是它的左上象限、右上象限、左下象限、右下象限的四分树编码。这样得到的四分树编码是一个二进制串。  在二进制编码开头添加一个1，再转化成十六进制便是该图像的十六进制四分树编码。  现在给定两个图像的十六进制四分树编码，问两图像的交的十六进制四分树编码。  十六进制四分树编码长度l1、l2<=100。 | 字符串处理+模拟  先将读入的十六进制四分树编码转化成二进制四分树编码，再构造出对应的两幅图像，求出交的图像后再将图像转化成二进制四分树编码，最后再转成十六进制四分树编码输出。  注意求交后4个象限内像素可能一样，要合并在一起。 | 时间：O(l1+l2)  空间：O(l1+l2) |
| 2003G | A Linking Loader | 给出一段程序，程序包括4种语句：定义一个变量的值、表明当前模块需要用到某变量、将一些数值或该模块声明过的某变量的值求十六位校验和、模块分界线。输出最终的校验和，以及被定义或声明的每个变量的值。  语句条数n<=1000。 | 模拟  先求出每个变量的值，再按语句顺序模拟操作，最后求出校验和以及每个变量的值，细节很多。 | 时间：O(n)  空间：O(n) |
| 2003H | A Spy in the Metro | 某城市只有一条双向地铁线，该地铁线有n个站点，告诉你相邻两站点列车行驶时间ai，以及所有列车从首站或末站出发的时间，某间谍0时刻在1号站点，他在t时刻要到n号站点见某人，假设换乘不需要时间，求出其在站点等待的最少时间。  2<=n<=50，0<=t<=200。 | 搜索  设p1[i][j]表示在j时刻是否有从首站开往末站的列车到达第i站，设p2[i][j]表示在j时刻是否有从末站开往首站的列车到达第i站，这个直接根据所有列车发车时间预处理即可。  接下来一遍做spfa即可，设f[i][j]表示第j时刻在第i站时最少的等待时间，转移如下：  若p1[i][j]=true，f[i+1][j+a[i]]=min{f[i+1][j+a[i]],f[i][j]}  若p2[i][j]=true，f[i-1][j+a[i-1]]=min{f[i-1][j+a[i-1]],f[i][j]}  f[i][j+1]=min{f[i][j+1],f[i][j]+1}  f[n][t]便是答案。 | 时间：O(knt)  空间：O(nt) |
| 2003I | The Solar System | 告诉你太阳系中某行星椭圆轨道半长轴a1、半短轴b1和环绕周期t1，另一个行星半长轴a2、半短轴b2，以及时间t，假设第二个行星从(a2,0)出发逆时针转，问其第t时刻所在坐标，假设太阳在x坐标非负整数的焦点上。  所有给出的数均是整数且在int范围内。 | 物理题+二分答案  开普勒第三定律：(t1/t2)^2=(a1/a2)^3，椭圆面积s=πab，太阳坐标(sqrt(a2\*a2-b2\*b2),0)。  先根据开普勒第三定律求而出第二个行星的环绕周期t2和太阳坐标，再求出时刻t时第二个行星在当前周期已经走过的面积占整个椭圆的多少。根据占的量与0.5的大小关系我们可以确定答案y坐标的正负，我们二分答案的x坐标，计算出此时坐标位置走过的面积占椭圆的量，与应该占的量比较，进一步二分直到精度满足要求为止。  精度要求极高，要用long double，而且eps至少1e-10。 | 时间：O(log(a2/eps))  空间：O(1) |
| 2003J | Toll | 已知一些城市和村庄之间的m条双向道路以及起点终点，你要从起点向终点运x单位货物，每经过村庄要交1单位，经过城市要交(x-1)/20+1单位，除了起点都要交，问最少需要多少单位货物才能满足要求。  1<=m<=1000，城市和村庄均至多26个，x<=1000。 | 图论  我们从终点向起点做最短路，设f[i]表示从i出发向终点运货至少f[i]单位，转移时根据当前位置是城市还是村庄可以求出转移的代价。 | 时间：O(522+m)  空间：O(522) |
| 2004E | Intersecting Dates | 已知n个日期的区间和另外m个日期的区间，求出所有区间满足其在后者的区间中但不在前者的区间中。  1<=n,m<=100。 | 模拟  将这n+m个区间[x,y]的端点日期x以及y加1后放在一起，按日期排序，然后从早到晚扫一遍，对于日期相同的一起处理。记录下经过该日期后被s1个n的区间包含，s2个m的区间包含，该日期前被s3个n的区间包含，该日期前被s4个m的区间包含，根据这些信息可以得知该日期是否是答案区间的左端点或右端点加1后的结果。在输出的时候右端点减1天再输出。 | 时间：O(n+m)  空间：O(n+m) |
| 2004H | Tree-Lined Streets | 平面直角坐标系上有n条街道（近似成线段），要在街道上种树，要求两棵树间距至少为50，且任意一棵树离十字路口（街道之间的交点）距离至少为25，问至多种几棵树。  n<=100。 | 计算几何+枚举  先求出每个街道的直线解析式，对于每条街道我们分别计算在该街道上能种几棵树。计算方式是求出所有别的街道与该街道交点，再对交点排序，可以算出相邻两个交点（或交点与线段端点或线段两端点）之间的距离len，由于树离十字路口距离至少25，那么对于每个距离其两端有多少交点就减几份25，剩下的部分就是可以种树的长度了。分别计算出答案后累加在一起便是最终结果。 | 时间：O(n2logn)  空间：O(n) |
| 2004I | Suspense! | 给出两幢楼房楼层数n、m以及间距d，每层均高3m，窗户高1.5m且比同一层最低点高1m。有两个人分别在两楼房顶楼，同时每层楼房都可能养鸟或猫。那两人要做一个模型桥，有一根类似抛物线的绳子在两人窗户间，其最低点再低1m便是桥面，桥面要比地面高1m，比他俩的窗户低2m。猫至多可以向上跳0.5m或向下跳3m，问使得猫吃不到鸟时最低桥面高度对应的绳子长度。  2<=n,m<=25，1<=d<=25。 | 数学题+枚举  由于所有给出的数值均是0.5的倍数，我们可以以0.5m为间隔枚举桥面高度，再判断是否可行。  判断时我们只要枚举每只猫能否上桥面、每只鸟能否被吃到，当只要有1只猫和鸟满足就是不可行的。  以左边房子的右下角为平面直角坐标系原点，根据两人窗户最低点的坐标和桥面高度，可以确定绳子的抛物线解析式。接着使用积分就能求出某段区间内的抛物线长度了。 | 时间：O(n\*(n+m))  空间：O(n+m) |
| 2005C | The Traveling Judges Problem | 给定一个n个城市m条带权双向道路图、k个人的起始城市和他们的终点t，当一些人在同一城市时他们可以统一行动（视为1人），问他们全部到达终点时走过的最短距离，同时给出方案（要求先满足经过的城市集合元素最少，再满足经过城市的集合元素字典序最小）。  1<=n<=20，1<=k<=10。 | 图论  本题可以利用斯坦纳树解决。  由于n很小，我们用邻接矩阵w[i][j]记录道路长度。设f[sta][i]表示由01状态sta代表的那部分人统一行动到达i时走过的最短距离，g[sta][i]表示对应的走过的城市的集合，u[sta][i]记录了当前状态从哪里转移过来。  则转移只有两类：1.将在同一点的两部分人合并，f[sta][i]=f[sta’][i]+f[sta-sta’][i]；2.一部分人移动到另一点，f[sta][i]=f[sta][j]+w[j][i]。第一部分我们需要枚举子集，时间O(3k)；第二部分我们则要做一次最短路，对于同一个sta，每次找到f最小的更新去其他f，时间O(n^2)。  f[2k-1][t]便是最短距离，再根据u可以求出方案。 | 时间：O(3k+2kn2)  空间：O(2kn) |
| 2005E | Lots of Sunlight | 水平线上有n幢房子，给定n幢房子的楼层数和相邻两幢间距，每层只有1间公寓，所有房子的公寓都视为宽w高h的矩形。若太阳完整照射到公寓的左边或右边或在该公寓所以在房子的上方，均视为被照射到。给出日出和日落的时间，求一些公寓被太阳照射到的时间段。  n<=100，一幢房子公寓数<=100，询问数<=1000。 | 计算几何  先根据日出日落时间计算出太阳在空中走1度经过的时间。对于每个询问找出该公寓所在位置，并与任意其他的房子最顶端进行一次计算（先算出高度差和水平距离差，用三角函数转成度数后再换算成经过的时间），再算出照射区间的交输出答案。  本题精度问题十分严重。 | 时间：O(nq)  空间：O(n) |
| 2005G | Tiling the Plane | 在平面直角坐标系中给出一个多边形，其周长为n，保证该多边形角均为直角且每条边长度均为整数，同时边必然与x轴或y轴平行。你可以复制无限个该多边形，并可以随便平移，但不能旋转或翻转，问能否覆盖整个平面。  n<=50。 | 图论+枚举  只有两种本质不同的铺满平面的情况：使用正四边形铺满平面（棋盘覆盖），或使用正六边形铺满平面（蜂巢覆盖）。因此一个多边形只有满足以下两个条件至少一个时才可以铺满平面：  1.在多边形边界上顺次存在四个点A,B,C,D（不一定要是多边形的顶点），使得A到B的边界与D到C的边界重合，B到C的边界与A到D的边界重合（棋盘覆盖）。  2.在多边形边界上顺次存在六个点A,B,C,D,E,F（不一定要是多边形的顶点），使得A到B的边界与E到D的边界重合，B到C的边界与F到E的边界重合，C到D的边界与A到F的边界重合（蜂巢覆盖）。  由于边长均是整数，我们选择的点必然是整点。我们先将多边形用一个长度为n的数字串表示出来，再复制一遍（环），再对于两种覆盖情况枚举AB位置（棋盘覆盖）或ABC位置（蜂巢覆盖），这样就可以确定剩余几个点的位置，相邻两点间的数字串便对应了多边形的边的情况。我们对于相对应的两段数字串，一个数字从前到后一个从后到前枚举，当数字代表方向正好相对时说明这一段是匹配的，如果全都匹配上了就说明重合了。只要满足上述两个条件之一便是可以覆盖的。 | 时间：O(n4)  空间：O(n) |
| 2005H | The Great Wall Game | 有一个n\*n的网格，棋盘上有n个石子，石子坐标是(ai.x,ai.y)，一次可以将一个石子移动到上下左右的一个相邻格子中，问最少的移动次数使得石子连成一条水平/竖直/斜的线。  1<=n<=15。 | 枚举+二分图最大加权匹配  我们先枚举n个石子可行的目标状态(bi.x,bi.y)，这总共有n\*2+2种。对于一种状态，一个石子从一点移动到另一点的代价是cost=|ai.x-bj.x|+|ai.y-bj.y|，我们令wij=-cost，这样就形成了一个二分图，用KM算法求出其最大加权匹配值再取反便是到达该目标状态的最少移动次数。对于所有的这些值求min便是答案。 | 时间：O(n4)  空间：O(n2) |
| 2005I | Workshops | 有n次会议m个房间，每个会议有开会的人数和持续的时间，每个房间有容纳最多的人数和开放的时间，会议要同时开始且一个房间只有一个会议召开，且要保证房间容纳人数≥会议的人数和开放时间≥会议持续时间，问在满足最多会议召开的前提下最多会议召开数和最少未能开会的人数。  n<=1000，m<=1000。 | 二分图最大加权匹配/贪心  本题可以使用KM算法解决，左边的点表示会议右边的点表示房间，若某会议可以在某房间召开，边权为该会议人数，不可以的话边权为-inf，求最大加权匹配即可，但这样需要常数优化十分麻烦。  本题可以也用贪心做，我们将房间按开放时间从小到大排序，从开放时间小的开始枚举，对于每个房间，找出还没被安排的会议中可以在当前房间进行的人数最多的会议，将其安排在该房间就行了，这样做就能得到最优解。我们可以证明不论怎么调整方案均不会变优。 | 时间：O(nm)  空间：O(n+m) |
| 2005J | Zones | 有n个信号站分别覆盖n个圆形区域，每个信号站范围内都有若干客户，同时告诉你所有公共信号区的信息（有k个），包括每个公共信号区内客户数量和哪些信号站覆盖该信号区，问恰好选m个信号站可以覆盖最多的客户数及方案。  m<=n<=20，k<=10。 | 枚举  由于n很小我们可以直接DFS查找，对于一个恰好选了m个信号站的情况，根据k个公共信号区的信息可以求出实际覆盖客户人数，记录下最优的方案就行了。 | 时间：O(2n\*kn)  空间：O(n+k) |
| 2006B | Remember the A La Mode! | 有n种饼片和m种冰激凌，已知每种饼片和冰激凌的数量ai和bi（保证饼片数量与冰激凌数量相等），以及任意饼片与冰激凌组合产生的利润wij（或不能组合），问让饼片和冰激凌组合在一起且都卖完的情况下最小和最大利润。  1<=n,m<=50，1<=ai,bi<=100，0<wij<10，wij为实数。 | 网络流  设(a,b)表示流量为a费用为b。  网络流建边如下：源向每个饼片建(ai,0)的边，每个冰激凌向汇建(bi,0)的边，每个饼片向每个冰激凌建(100,wij)的边（可以组合时），所有边同时建好对应的反向边。分别求一遍最小费用最大流和最大费用最大流边是最小和最大利润。 | 时间：O(knm(n+m))  空间：O(nm) |
| 2006D | BipartiteNumbers | 给定一个数n，求出最小的二段数满足其比n大且是n的倍数。二段数满足恰好包含两种不同的数字s和t，s不是0，并且所有s均在所有t前面。有case组数据。  1<=n<=99999，1<=case<=200。 | 数论+枚举  我们可以枚举s和t的值，由于循环节长度<=n，s和t的位数均<=n。我们先枚举有几个s，统计出所有f[i]表示最少多少个s时对n取模为i。再枚举有几个t，当有j个t时，记录下当前的模值a以及若t前的s模为1时此时的模值b，用扩展欧几里得求出(a+bx)%n=0时x可以的取值，此时的f[x]就对应了需要的最少的s个数。  对于所有可行的方案我们求出最小的满足条件的数，注意该值要比n大，而且数据组数较多需要剪枝。 | 时间：O(case\*n\*102)  空间：O(n) |
| 2006E | Bit Compressor | 有一种压缩二进制串的方法是这样的：　　将连续的n个1替换为n的二进制表示（替换发生当且仅当这种替换减少了二进制串的总长度且连续n个1左右必须是0或者是串的开头、结尾）。这种方法的弊端在于有时候解压缩算法会得到不止一个可能的原串。你需要判定能否利用压缩后的信息来确定原串。已知原串长m，原串中1的个数n，以及压缩后的串。  1<=m<=131072，压缩后串长len<=40。 | DP  原串0的个数为m-n，当m-n>len时不存在原串。令m-=n，则m现在表示0的个数，设f[i][s1][s2]表示做完压缩串的第i位，原串已经有了s1个0、s2个1时的方案数。当i+1位是0时，f[i+1][s1+1][s2]+=f[i][s1][s2]；当i+1位是1且i位是0或开头结尾时，枚举j，记录下i+1位到j位的二进制数值为s，当j+1位是0或开头结尾时，f[j][s1][s2+w]+=f[i][s1][s2]，w由s决定：当s>3时w=s，当s=1时，w=1，当s=3时，w=2或3。初始时f[0][0][0]=1，根据f[len][m][n]的值为0、1或大于1确定原串是不存在、存在且唯一或存在但不唯一。  但这样会爆空间，而不论压缩串为多少，对于f[i]来说，f[i][s1][s2]非0的个数很少，因此我们可以记录下哪些状态非0，只用这些状态来转移，时间也将远远低于理论复杂度。 | 时间：O(len2n(m-n))  空间：O(10000len) |
| 2006G | Pilgrimage | 有若干个人去旅行，有一个人从开始就在队伍中直到终点，由他处理全队的财产收支，途中有4种可能发生的情况，1.花费了一些钱，2.每个人都捐出同样的钱数给他保管，3.有若干人加入队伍且每人交了当前的人均公共财产的钱数，4.有若干人离队且每人取走当前人均公共财产的钱数。但巧合的是每次计算人均公共财产时均为整数，问最初可能的人数。  总的情况数m<=50，花费钱数k<=2000，数据组数t<=30000。 | 数论+模拟  首先我们可以发现情况2不会对答案造成影响，可以忽略。由于我们只有情况3和4要计算，我们可以以情况3和4为分界线，中间的情况1都累加在一起处理。设初始人数为x，根据情况3和4我们可以计算出任意时刻的人数，由于计算时结果要是整数，我们对于每一段区间都计算出当前这一段花费钱的和的所有约数，再转化到初始人数上，将每段的答案求交集便是最终的答案，注意要保证任意时刻人数均至少为1。由于初始资金未知，在第1个情况3或4出现前的花费和可以忽略；同样的，最后1个情况3或4出现后的花费和也可以忽略，因为之后不再计算。本题无限解的情况是忽略开头和结尾的花费后中间没有任何花费。 | 时间：O(tm\*sqrt(mk))  空间：O(sqrt(mk)) |
| 2006I | Degrees of Separation | 给定n个点m条边的无向图，问图中最远两点的距离。  1<=n<=50。 | 图论  对原图求一遍floyd就能算出两点间距离，输出最大的即可。 | 时间：O(n3)  空间：O(n2) |
| 2006J | Routing | 给定一个n个点m条有向边的图，求最少标记的点的个数使得存在1到2和2到1的路径只经过标记点。  1<=n<=100，m<=min(1000,n\*n) | 图论+DP  先进行一遍floyd求出所有的w[i][j](1<=i,j<=n)，w[i][j]表示从i到j的最短路径长度。  再进行DP，设f[i][j]表示存在1到i和j到1的路径时最少标记点个数，初始时f[1][1]=1，我们用spfa去求出所有的f[i][j]，而f[i][j]可以转移到f[k][j]、f[i][k]、f[j][i]。  转移状态如下：  1.f[k][j]=min{f[i][j]+w[i][k]-(k==j)}(当k!=i)。  2.f[i][k]=min{f[i][j]+w[k][j]-(k==i)}(当k!=j)。  3.f[j][i]=min{f[i][j]+w[i][j]-1}(当i!=j)。  f[2][2]便是所求的答案。  设spfa的常数为k。 | 时间：O(n3+kn2)  空间：O(n2) |
| 2007A | Consanguine Calculations | 每个人都有ABO血型和Rh血型，每种血型都有两个该血型的等位基因，给出父母和孩子三者其中两人的血型，求出剩下的那个人的血型。  ABO血型包含A、B、O、AB，Rh血型包含+、-。 | 枚举  由于血型的情况总数很少，我们只需要直接枚举每个人的4个等位基因再判断是否满足条件就行，然后根据4个等位基因便可以求出那个未知的人的血型了。  注意字符串处理的细节。 | 时间：O((32\*22)3)  空间：O(1) |
| 2007G | Network | 有n条信息，每条信息被划分成若干个信息包，共有m个信息包。给出信息包到达的顺序，有一个缓存区存放信息包，每次可以选择让当前信息包直接通过或放入缓存区或从缓存区选择一个通过，要求同一条信息的信息包按顺序连续通过，信息通过顺序随意，问缓存区最小是多少。  1<=n<=5，1<=m<=1000，信息包大小<=64。 | 枚举+模拟  先枚举n条信息通过的顺序，然后再按照信息包到达顺序进行模拟，先将当前信息包放入缓存区，再将当前要通过的信息编号的在缓存区中的可以通过的信息包尽量多的通过，再记录下此时缓存区内信息包总量，求出最大值便是当前信息通过顺序下缓存区最小大小，就出这些数的最小值便是答案。 | 时间：O(2nm)  空间：O(64nm) |
| 2007I | Water Tanks | 有n个水箱放成一排，相邻两个之间有水平的管道连接，保证管道高度逐渐递增且不超过其相邻两水箱高度，1高度的水等于0.097标准大气压，水箱横截面积均为1，问不断向1号水箱加水最多可以加多少水。  1<=n<=10，水箱及管道高度均不超过1000000。 | 物理题+递推  从1号水箱开始计算有多少水，水的高度有三种状态：低于左边管道，介于左右管道间，高于右边管道。  而低于左边管道时，所有右边水箱、当前水箱上部、左边水箱上部都连通；介于左右管道间时所有右边水箱、当前水箱上部连通；高于右边管道时，右边水箱和当前水箱上部分开。关键是要处理好每种状态下每部分的空气压强，然后根据连通器同一高度压强相等，就可以求出每个水箱内的水量了。  要注意水箱高度与其右边管道高度一致的情况，此时该水箱不存在第三种情况，容易误判。 | 时间：O(n^2)  空间：O(n) |
| 2007J | Tunnels | 给定一个n+1个点m条边的无向图，1号点有一个人，0号点是出口，你可以炸毁一些边让他永远走不到出口，求最坏情况下要炸毁的边数。  n<=50，m<=1000。 | 网络流+DP  显然本题不是直接求1到0的最小割，因为可以等那个人走了几步再炸或根据他走的策略炸。  设f[i]表示从i出发时阻止他到0的最少炸毁边的数量，我们先求出1-n号点到0的最小割，显然这是一开始就先炸毁边的答案，其实我们可以先让他走到一些点再炸。于是得到以下做法：找出还在图中的点的f[i]的最小值k，将所有等于最小值的点删除，对于剩下的点再做最小割得到值为f[i]’，则f[i]更新为min(f[i],f[i]’+k)。这相当于是说如果他走到这些点我们可以直接堵死，如果不走到我们便可以忽略这些点做最小割将他从i到0的路堵死，同时这也满足递推的性质。我们只要这样操作直到除0以外没有别的点在图中为止，此时f[1]值就是答案。  本题看似有O(n^4m)的时间，但由于网络流速度几乎达不到n^2m、每次删点不一定只有1个、删了点后点和边数变小网络流更快，所以不会超时。 | 时间：O(n4m)  空间：O(n+m) |
| 2008A | Air Conditioning Machinery | 在一个长方体空间内装管道，可以将长方体空间视为由n\*m\*k个单位正方体构成的东西，你有6根相同的L型弯管，每根均占用4个单位正方体空间，且只有两个接口，只有接口对应才算接通，给出长方体空间中需要连接的两个接口位置，问最少需要多少弯管才能接通两个接口。  1<=n,m,k<=20。 | 搜索  本题数据范围很小，直接DFS搜索即可，从一个接口出发，不断改变弯管方向尝试加入一根弯管，如果可行就加入，我们发现对于1个接口可以往上接的弯管最多8种形状，直到到达目标接口为止，注意当前弯管不能与之前的弯管重合。 | 时间：O(86)  空间：O(nmk) |
| 2008B | Always an Integer | 给定一个自变量为n的多项式，问n为任意正整数时该多项式是否均为整数。多项式一定是(P)/D的形式，P是由若干单项式±Cn^E组成的且是降幂排列，C、D、E均是非负整数。  0<=E<=100，0<D<231，0<=C<231。 | 字符串处理+枚举+数论  首先处理好多项式P，将其转化为n对数(ai,bi)表示其含有ain^bi项。  设次数最高为m，可以证明我们只需枚举n为[1,m+1]内的所有整数并判断P(n)%D是否为0即可。 | 时间：O(nm)  空间：O(n) |
| 2008F | Glenbow Museum | 有一个有n个角的直角多边形，有90°（R）和270°（O）两种角，在保证其内部存在一个点可以看到整个图的条件下，有多少角的序列存在对应的多边形符合条件。  n<=1000。 | 数学题  由于该多边形内部存在点可以看到整个多边形，我们可以将其放到一个平面直角坐标系中，原点可以看到整个多边形内部，则每个象限都是ROROR……OR交替出现角，我们设每个象限有x个角，y个O角，则x1+x2+x3+x4=n，2y1+1+2y2+1+2y3+1+2y4+1=n，可得y1+y2+y3+y4=(n-4)/2，所以n为奇数或小于4时无解，有解的话我们枚举y1，对于y2,y3,y4我们直接用组合数求解即可，由于在y1确定的情况下，所有2y1+1个角都可以作为序列起始点，因此还要乘上2y1+1。设(n-4)/2=m，则答案为。 | 时间：O(n)  空间：O(1) |
| 2008G | Net Loss | 给定一个n次函数f(x)和直线x=x0。求出两个一次函数y1=a1x+b1和y2=a2x+b2，满足在[-1,x0]内g(x)=y1，在[x0,1]内g(x)=y2，且当x=x0时y1=y2，使得(f(x)-g(x))^2在[-1,1]内的距离最小。距离的定义为差值的平方的定积分。  1<=n<=10，-1<x0<1，函数的系数均在[-1,1]内，且x0至多3位小数，答案要求3位小数的精度。 | 数学题+三分  最朴素的做法是枚举在x=x0上y1和y2的交点，再对于y1和y2分别枚举函数解析式，找出其中最优的。  而观察数据可得，当交点确定时，距离值与函数的斜率呈单峰关系，我们可以三分斜率找出最优方案。同时，距离值与交点纵坐标y0也呈单峰关系，我们同样可以三分纵坐标。  因此本题我们可以先三分纵坐标，再对两个函数分别三分斜率，找出最优解。  注意本题对精度要求很高。  设三分区间长度为l1和l2，精度为eps1和eps2。 | 时间：O(n2\*log(l1/eps1)\*log(l2/eps2))  空间：O(n) |
| 2008J | The Sky is the Limit | 在地平线上有n座山，山都可以看做二维平面上两边长度相等的三角形。给出山脉的最高点到某个固定的点的水平距离，山脉的垂直高度和山脉的底边的宽度。求出这些山脉轮廓线的长度。  1<=n<=100。 | 计算几何  我们用扫描线的方法解决本题。总共有2n条线段，先找出所有这2n条线段的交点的x坐标，以及初始时的山顶和左右山脚的x坐标，将这些坐标从小到大排序，则在相邻两坐标之间只有一条线段在最高处，直接计算该线段的长度并加在一起就是答案。 | 时间：O(n3)  空间：O(n2) |
| 2009A | A Careful Approach | 有n架飞机，每架飞机有可以着陆时间的区间[ai,bi]，求飞机着陆时间间隔的最小值最大是多少。  2<=n<=8，0<=ai<=bi<=1440。 | 搜索+二分答案  我们二分间隔的最小值，再枚举飞机着陆的次序，判断该间隔是否存在可行方案，直到精度满足要求为止。  设二分的区间长度为len，二分精度为eps。 | 时间：O(2n\*log(len/eps))  空间：O(n) |
| 2009F | Deer-Proof Fence | 有n棵树，每棵树都有一个坐标(xi,yi)，现在要用栅栏将树都围起来，使得树离栅栏距离至少为r，问栅栏的最短长度。  1<=n<=9，-100<=xi,yi<=100，0<r<=200。 | 计算几何+状态压缩DP  对于一些树在一个栅栏中的情况，我们需要的栅栏长度为这些树位置的凸包周长加上半径为r的圆周长。  设f[sta]表示已经将二进制状态sta对应的树用栅栏围住时的栅栏最短长度，枚举其补集的所有子集sta’，按上述做法求出sta’对应的树需要的栅栏长度len，然后进行转移：f[sta+sta’]={f[sta+sta’],f[sta]+len}。  f[2n-1]便是答案。 | 时间：O(3nn2)  空间：O(2n+n) |
| 2009H | The Ministers' Major Mess | 有n个议案和m个大臣，每个大臣会投si票，每票会支持或反对一个议案，要满足每个大臣大于一半的投票。问是否有解，若有解输出每个议案是一定要通过、一定要否决还是可以通过或否决。  n<=100，m<=500，1<=si<=4。 | 图论+枚举  由于每个大臣要满足大于一半的票，即最多1票不成功，即选择两个同一大臣的投票至少要满足其中1票，这可以转化成2-SAT问题。构造一个包含2n个点的图，共n对点，每对代表了一个议案通过还是否决，显然任意1对点不能同时选。  若一个大臣si<=2，则两票必须都通过，我们将这两票对应点的对立点标记成false；若一个大臣si>2，枚举其si票中的两票x和y，当x!=y时，构建一条从x对立点到y的有向边，表示选择了x的对立点的方案，为了满足该大臣的要求必须要选择y。  枚举2n个点，对于每个点做一遍BFS，求出其能到达的所有点（包括自己），若其到达了一个本身是false的点或者到达了某个点及其对立点，则说明不能选择该点对应的情况。根据这2n个点能否选择的结果，可以确定本题要求的答案。 | 时间：O(nm)  空间：O(n+m) |
| 2009I | Struts and Springs | 平面上有n+1个矩形窗户，边均是水平边或竖直边，保证矩形之间只有包含关系。除了最外面的窗户，每个窗户都有6根支杆或弹簧，分别连接该窗户两水平边、两竖直边和直接包含它的窗户的相对边之间。支杆长度不变，弹簧则按最初的比例放缩。若竖直方向或水平方向的3根都是支杆则最上或最右支杆变成弹簧。给出最大窗户的宽和高，其余n个窗户的宽、高、左上角相对于最大窗户左上角的坐标以及它们的6根是支杆还是弹簧，问当最大窗户宽和高变成X和Y时n个窗户的宽、高和左上角坐标，共m个询问。  n<=500，m<=500，最大窗户宽和高均不超过1000000，保证操作后任意距离均为整数。 | 递推  先对于竖直或水平方向均是支杆的情况进行处理，只要将需要改的那根直接改成弹簧即可。同时询问之间也是相互独立的，可以分开算。  接着我们做一遍拓扑排序，将窗户从外到内排序，使得我们可以根据外围的窗户求出其内部直接包含的窗户的状态。  然后我们从外到内递推，根据外围窗户状态的变化以及当前窗户的6根是支杆还是弹簧，可以计算出当前窗户的状态。  对于每个询问分别计算即可，注意计算过程中会爆int而且常数较大。 | 时间：O(nm)  空间：O(nm) |
| 2010B | Barcodes | Code-11是条形码的一种编码方式，被编码的字符被限制为0-9、'-'或一种特殊的符号（代表开始和结束），一个字符会由5个相邻的区域来编码，每个区域有宽窄之分（题目给出了所有字符的宽窄编码），宽区域是窄区域的2倍宽度，相邻两个字符之间有一个窄区域。在一个合法的Code-11编码中，开头和结尾都有1个特殊符号，中间部分是正常字符，并设有两个检验字符。给出n个区域的读取宽度（与实际宽度有至多5%误差），区域可能是倒着读取的，问该区域能否解码以及能解码时检验字符的正确性。  n<=150，读取宽度<=250且是正整数。 | 模拟  首先是要确定读入的区域中哪些是宽区域哪些是窄区域：先在整数范围枚举标准宽度，处理掉一些情况，若没找到，则区域宽度必然不会很小，我们可以确定宽区域和窄区域的读取范围，接着通过实数运算确认是否存在实数长度的标准宽度，若没有就不合法。  然后分别处理正着读取和反着读取（将区域翻转即可）的情况：首先若区域个数模6非5为不合法，接着可以5个一组确定每个字符，档开头和结尾非特殊符号时不合法、任意位置出现不可读字符时不合法、中间部分出现特殊符号时不合法、两组的间隔用宽区域时不合法，其他情况下合法，最后判断两个检验字符的正确性。 | 时间：O(n)  空间：O(n) |
| 2010C | Tracking Bio-bots | 有一个n行m列的网格房间，共有k堵墙，墙都只占了一行若干列网格。有一个机器人，当它在x行y列时它只能向x+1行y列或x行y+1列移动，问这n行m列的网格中（不包括墙）有多少个格子使得机器人以该点为起点不能走到n行m列的格子。  1<=n,m<=1000000，0<=k<=1000。 | DP  由于k很小，我们可以先离散化，用一个位置代表原来的一段区间，这样房间就至多2k+1行2k+1列了。在确定哪些格子是墙后，我们可以直接进行DP，确定哪些格子不能到达(2k+1,2k+1)，将所有这样的格子代表的原来的格子数加起来便是答案。 | 时间：O(k2)  空间：O(k2) |
| 2010D | Castles | 有一个城堡群，共n个城堡，有n-1条无向边将城堡群联通（即一棵树），你可以选择任意一个城堡作为初始进攻点，每个城堡都有攻击需要的最少人数、攻击会阵亡的人数、占领后需要驻守的人数，你的部队在攻下某城堡后可以继续进攻相邻的城堡，同时你的部队最多沿着某条边走两次，问攻克所有城堡的最少部队人数。  1<=n<=100。 | 树形DP+枚举  攻击阵亡的人数和占领后驻守的人数本质是一致的，可以加在一起算。  先枚举初始进攻点，以其为根做一遍树形DP，对于每个点i，我们设f[i]表示攻克以i为根的子树的所有城堡最少部队人数，g[i]表示攻克以i为根的子树的所有城堡还有多少人返回到i城堡（因为有些城堡攻克后还有人不必阵亡或驻守），h[i]表示攻克以i为根的子树的所有城堡阵亡和驻守的总人数。则g[i]=f[i]-h[i]，h[i]可以直接将所有儿子的h值和本身消耗的人数加在一起得出。对于f[i]的计算，关键是要确定攻击其儿子对应的子树的顺序，我们发现由于对于任意k值f[k]>=g[k]，我们只要将i所有儿子按g值从大到小排序，按这个顺序依次攻击那些子树的所有城堡便是最优策略，于是我们就算出了不包括i点本身的f[i]值，然后将f[i]再加上本身的阵亡和驻守人数，同本身的最少攻击人数求max后，便求出点i的f值了。  最后将每个初始进攻点的f值求max便是最少的部队人数。 | 时间：O(n2logn)  空间：O(n) |
| 2010G | The Islands | 有n个岛屿，每个岛屿都有一个坐标(xi,yi)，两岛屿距离为((x1-x2)^2+(y1-y2)^2)^0.5且岛屿横坐标递增。某人要从1号岛屿到n号岛屿再回到1号岛屿，要求满足2到n-1号岛屿均恰好经过一次，同时有两个特殊点b1和b2，要求它们分别在两条路径上。问最短路径长度。  4<=n<=100，0<=xi,yi<=2000，1<b1,b2<n。 | DP  从1到n再到1等价于两条1-n的路径。  设f[i][j]表示两条路径分别是1-i和1-j时最短的路径长度，w[i][j]表示岛屿i到j的距离，令k=max{i,j}+1，则有转移：f[k][j]=min{f[k][j],f[i][j]+w[i][k]}，f[i][k]=min{f[i][k],f[i][j]+w[j][k]}，当k==b1时不能进行后者的转移，当k==b2时不能进行前者的转移，初始时f[1][1]=1，当f[i][j]中i==n||j==n时转移到f[n][n]，f[n][n]便是答案。 | 时间：O(n2)  空间：O(n2) |
| 2010J | Sharing Chocolate | 把一块n\*m的矩形巧克力分成k块，给出每块的大小，要求恰好分完且每块都是矩形，你每次可以将巧克力在某行或某列分开成两块但不能直接掰出一小块，问是否有方案满足要求。  1<=n,m<=100，1<=k<=15。 | DP  假设n>=m，当k块和不是n\*m时显然无解。  由于每块巧克力必须沿着某行或某列分开，我们可以设p[sta][i]表示选择巧克力的二进制状态是sta，能否构成其中一边是i的矩形，tot[sta]表示巧克力的二进制状态是sta时选择的所有巧克力的大小之和。设sta1是sta的一个真子集，枚举sta1，令sta2=sta-sta1，则p[sta][i]|=p[sta1][i]&&p[sta2][i]，根据tot[sta]可以计算出另一边的边长。而且i≤n，否则也没有意义，同时对于i状态我们还要压位处理，不然会超时。  本题对常数要求很高。 | 时间：O(3k\*(n/32))  空间：O(2k\*n) |
| 2011A | To Add or to Multiply | 给定两个数a和m，以及起始区间[k1,k2]和目标区间[t1,t2]，你每次可以选择将原区间加a变成[a+k1,a+k2]或乘m变成[mk1,mk2]，问最少操作步骤的方案使得新区间被目标区间包含，有多解时输出字典序最小的。  1<=a,m,k1,k2,t1,t2<=1000000000，k1<=k2，t1<=t2。 | 数学题+枚举  当[k1,k2]∈[t1,t2]时操作为空，否则先计算我们至多需要乘的次数tot，当m=1时tot显然为0，当m>1时枚举一下就能求出tot。  接着我们枚举乘的次数，根据乘的次数可以算出若加操作均在最后的乘操作后可以操作的次数的区间[l,r]，当该区间为空时即当前乘的次数无解。接下来就类似于求一个m进制的数在该区间中且各位之和最小（f[0]值可以超过m），设f[i]表示在第i个乘操作后加操作的次数，f[0]是在第一个乘操作前的加操作次数。因为还要满足字典序最小，我们从f[0]开始计算，求出最小的f[i]使得当前这个m进制的数≥l，如果这时数＞r则使得f[i]减1，这样就能求出当前乘的次数下操作步骤最少且字典序最小的方案了。找出所有方案中最优的便是答案。 | 时间：O(tot2)  空间：O(tot) |
| 2011C | Ancient Messages | 给定一个n\*m的像素图（0表示白色1表示黑色），图上有若干象形符号，每个象形符号均连通且每个黑色点必属于一个象形符号，象形符号共6种如下图，要求区分所有象形符号，求出图上每个符号的类型。  http://oj.tsinsen.com/RequireFile.do?fid=a8AgqBbj  1<=n,m<=200。 | BFS  我们发现6种象形符号的区别是其内部包含的白色区域个数，它们分别有1、3、5、4、0、2个白色区域，因此我们可以先BFS找出所有黑色的连通块，再BFS找出每个白色连通块以及它被哪个黑色连通块包含（或不被任意一个包含），根据每个黑色连通块内白色区域个数我们就可以确定它是哪个符号。 | 时间：O(nm)  空间：O(nm) |
| 2011H | Mining Your Own Business | 给出一个m条边的无向连通图结构的矿井，要在某些点设立竖井便于逃生，使得不论哪个点被破坏时剩余点都依然还存在到达竖井的路。问至少设立几个竖井和设立竖井的方案数。  1<=m<=50000。 | 图论  设点数为n，因为m>=1，所以n>=2，而图是连通图，则n<=m+1。  当破坏的点在非割点位置时都只需至多两个竖井（一个竖井时破坏点恰好在竖井位置就不满足要求了），而当破坏的点在割点位置时，将原图缩点后所有叶子结点都需要一个竖井。  因此我们要先求出原图的所有割点，对于每个由非割点构成的连通块，若其只与一个割点相连，该连通块就是缩点后的叶子结点。竖井设立个数即叶子结点个数，方案数即叶子结点对应非割点连通块大小的乘积。  注意一种特殊情况：当原图所有点是一个双联通分量时，依然需要设立2个竖井，方案数为C(n,2)。 | 时间：O(n+m)  空间：O(n+m) |
| 2011I | Mummy Madness | 有一个无限的正方形网格，初始时你在(0,0)，有n个木乃伊，它们在(xi,yi)，每次你可以移动到周围8个格子中的一个，木乃伊也是如此。在每个单位时间内你先移动再木乃伊移动，若你与任意木乃伊在同一格则你被抓住。问你最多活多少时间。  n<=100000，|xi|,|yi|<=1000000。 | 二分答案+高级数据结构  设二分的t最大为m，m可以定为所有|xi|、|yi|的最大值。  二分活的时间长度t，在t时刻人和木乃伊能走到的范围均是一个边长2t+1的正方形，我们只要判断n个木乃伊正方形能否覆盖人的正方形就可以了，这可以用扫描线的方法解决，在所有木乃伊正方形覆盖状态变化时都判断一下能否覆盖住人的正方形即可。由于正方形边长一定，还可以简化程序：1.在任意时刻，n个正方形覆盖状态变化时的加入边和删除边的顺序总是确定的，只需按横坐标排序一遍即可；2.在任意时刻，每个正方形不能覆盖的人正方形的在y坐标上宽度也是一定的，即|yi|。因此我们只需记录下做到当前这条竖线时，离x轴最近的两个点（一上一下），其差值与二分的时间t比较便能知道当前在这条线上能否覆盖住人的正方形了，这个用树状数组维护即可。 | 时间：O(nlognlogm)  空间：O(n+m) |
| 2011J | Pyramids | 有两类金字塔，分别是xH和xL，xH代表了有x层每层用了x^2、(x-1)^2、…、1^2块砖的金字塔，xL代表了有(x+1)/2层每层用了x^2、(x-2)^2、…块砖的金字塔。现在有n块砖来搭若干座金字塔，方案要满足砖被全部用完、金字塔两两不同、金字塔至少2层，当存在方案时输出金字塔数最少的方案，当有多个满足时输出用砖最多的金字塔最大的，还有多个满足时输出用砖次多的金字塔最大的，以此类推。  1<=n<=1000000。 | 搜索/DP+数学题  1H、1L、2L是不合法的金字塔。  先枚举x的值找出所有合法的且用砖量<=n的金字塔，并将它们按照用砖量从大到小排序（相同时H金字塔比L金字塔在前面）。这里存在一个结论：有解时至多选了6座金字塔，因此我们可以使用搜索剪枝或者DP来解决该题，理论时间复杂度很高但很快，注意记录最优的方案。  设合法金字塔数为m，m约为sqrt(n)。 | 时间：搜索：O(m6)，DP：O(6mn)  空间：O(n) |
| 2011K | Trash Removal | 给定一个n个顶点的多边形，求出最小的距离d使得可以用距离为d的两条平行直线夹住该多边形。  3<=n<=100。 | 计算几何  凹多边形是毫无意义的，我们先对该多边形顶点求凸包组成新的多边形。为了夹住该多边形，其一边必然在其中一条直线上，因此可以枚举是哪条边，算出其他所有顶点到该边所在直线距离的最大值，便是该边在直线上时最小的距离d。对于所有边的结果求min便是答案。 | 时间：O(n2)  空间：O(n) |
| 2012B | Curvy Little Bottles | 给定一个n次的多项式，有一个瓶子是该多项式在x∈[xlow,xhigh]内部分的曲线绕x轴旋转一周而成的，同时有一个增量m表示两个标记间的瓶子体积，问该瓶子体积是多少，以及从xlow开始标记（不算xlow）的前8个（或不够）标记到xlow的距离。  0<=n<=10，-100<=多项式系数<=100，-100<=xlow<xhigh<=100，1<=m<=500。 | 微积分+二分答案  设多项式为f(x)，则瓶子截面积的函数为g(x)=πf(x)^2，瓶子的体积为，对于标记我们可以二分标记的值，用积分求出该标记到前一个标记这段区间的体积，当该体积恰好为m时便是标记的位置。  设二分的精度为eps。 | 时间：O(n2+log(1/eps)\*n)  空间：O(n) |
| 2012C | Bus Tour | 一个城市的交通可以抽象成有n个点m条带权无向边构成的连通图，其中1号点是汽车出发点，n号点是景点，2至n-1号点都是旅馆，每个旅馆都有客人上车。汽车从1号点出发载上所有旅客到n号点，再将旅客送回各自旅馆并返回1号点。现在要求前(n-2)/2个旅馆上车的客人也必须是前(n-2)/2个下车的，问满足条件时汽车行驶最短路径的长度。  3<=n<=20。 | DP+搜索+图论  先用floyd计算出任意两点之间的最短距离。  设f[sta][i]表示从1号点出发已经载的客人对应旅馆的二进制状态为sta且当前在i号点，f2[sta][i]表示从n号点出发已经载的客人对应旅馆的二进制状态为sta且当前在i号点。对于某个状态f[sta][i]，我们可以枚举sta中某个为1的点j，表示我们从之前的某个不经过i的点且其当前在j点的状态，可以通过它转移到f[sta][i]。其中sta中包含超过n/2+1个点的状态不必计算。  在计算出所有f[sta][i]后我们DFS枚举所有恰好选择(n-2)/2个旅馆和1号点的情况sta1，根据sta1我们可以计算出对应的包含n号点和剩余旅馆的状态sta2，同时计算出我们返回时包含前(n-2)/2个旅馆和n号点的状态sta3，1号点和剩余旅馆的状态sta4，枚举sta1到sta2会先到达的点和sta3到sta4会先到达的点就可以算出当前状态的最短路径长度，求出它们的最小值就是答案。 | 时间：O(2nn2+C(20,10)\*n2)  空间：O(2nn) |
| 2012E | infiltration | 有一个有向完全图，定义一个点控制的点为其自己和其直接指向的点，求出选择至少多少个点才能控制所有点。  点的个数<=75。 | 枚举+图论  本题存在一个结论：设点数为n，一个有向完全图至少存在一个点出度>=n/2，用反正法易证明其正确性。  而n<=75，我们每次删掉图中出度最大点的所有控制点和有关的边，直到全部删光，由于一次删至少n/2+1个点，因此答案不超过6。  我们可以先枚举至多选5个的情况，有解就输出最优的，否则按上述方法操作就能解决本题，注意状态合并要压位。 | 时间：O(C(n,5))  空间：O(n2) |
| 2012L | Takeover Wars | 有两个公司x和y，x公司有n个子公司，y公司有m个子公司，每个子公司都有市场价值。x和y公司轮流操作，x公司先，有两种操作：合并当前公司的2个子公司，新子公司市场价值为两者的和，原来的2个子公司消失；当前公司1个子公司消灭对方的1个子公司，要求当前的子公司市场价值大于对方，结束后当前子公司市场价值不变，对方子公司消失；若上述两个都不行就不操作，否则必须选择其中一个执行。若一家公司子公司个数变成0就输了，问最后谁能获胜，保证双方不会在任何时刻拥有同价值的子公司。  1<=n,m<=100000，子公司价值<=10^12。 | 贪心+模拟  首先将两家公司的子公司按价值从大到小排序。  首先有4个结论：如果要消灭对方，必然是我方最大价值子公司消灭对方最大价值子公司；如果合并，必然是最大和第二大价值的子公司合并；如果对方合并操作后，我方仍然可以执行消灭操作，我方必然可以一直执行消灭操作；如果我方执行消灭操作，对方必然只能选择合并操作。  因此我们只需枚举x公司第一步执行的操作（二选一），之后双方的操作必然是确定的，直接模拟一遍就知道最后谁获胜了。 | 时间：O(nlogn+mlogm)  空间：O(n+m) |
| 2013A | Self-Assembly | 有n种正方形分子，每种都有无限个，且分子可以任意旋转翻转。其四条边上都有一个标记，标记可以是大写字母+正负号或者00（如A+，Z-，00等），其中大写字母相同且正负号相反的边可以连接在一起，使得两个分子拼接在一起，00边不与其他任何边连。问这些分子能否拼出无限的结构体。  n<=40000。 | 搜索  我们发现虽然分子种类多，但边的种类很少，只有53种且00边没有用。于是我们可以构造出一个只有52个点的图。对于是否是无限的结构，我们只要关心能否找到一个环就行，而由于可以旋转翻转，必然有方案分子不会重合。  设g(x)为与x可以连接的边的标记，如g(A+)=A-，对于每个正方形，如果有两条不同边其标记非00，就添加有向边(x,g(y))和(y,g(x))。如果图中有环就是无限的结构。  求环的话我们可以枚举每个点，从它出发做BFS，如果会回到它本身就说明有环。 | 时间：O(16n+523)  空间：O(522) |
| 2013B | Hey, Better Bettor | 在一个赌场你可以赌无限次，在赌博结束后如果你赚了则可以获得所有钱，如果你亏了可以把损失的x%还给你。每次赌博要1元，赢了就返还2元，输了就不返还。给出x和赢的概率p%，问在最优策略下最大期望收益。  0<=x<100，0<=p<50，且至多两位小数。 | 概率+数论  令x=1-x%，p/=100，p=0则答案显然为0。  有一个明显的DP：设f[i][j]表示赌了i次赢了j次的概率，递推若干次后可以得到较优解，但显然还不够。  而我们发现其实只要关心当前赚或亏了多少钱。用f[i]表示当前赚了i元的最大期望收益，明显的策略是当亏a元或赚b元时停止赌博否则继续。则-a<i<b时，f[i]=p\*f[i+1]+(1-p)\*f[i-1]，即f[i+1]=1/p\*f[i]+(1-p)/p\*f[i-1]，f[-a]=-xa，f[b]=b，f[0]便是答案。  令g[i]=f[i-a]，根据之前转移式可得特征根方程px^2-x+(1-p)=0，根据0<=p<0.5可知存在两个特征根x1=(1-p)/p和x2=1，因此g[i]=c1\*x1^i+c2\*x2^i，再g[0]=-xa、g[a+b]=b即可求出c1,c2，接着就求出了g[a]的值。g[a]=(a^a-1)\*(ax+b)/(a^(a+b)-1)-ax。  而对于a和b值的确定，我们发现当a确定时g[a]关于b为单峰函数，而a关于g[a]也是单峰函数，可以用log^2(length)时间完成二分操作。 | 时间：O(log2(limit))  空间：O(1) |
| 2013C | Surely You Congest | 有一个n个点m条无向边的道路图，边有权值代表了人通过该路的时间，有k个人要从他们各自的起点出发到1号点去，所有人都同时开始行动且只走最短路，若某时刻两个人都在同一点进入同一边就会造成交通堵塞，问在避免堵塞的情况下最多有多少人可以到达1号点。  1<=n<=25000，1<=m<=50000，1<=k<=1000。 | 图论+网络流  先从1号点开始做一遍最短路，求出它到所有点的最短距离f[i]，对于某条连接x和y的边v若f[x]+w[v]=f[y]，则说明在y的人可以通过这条边到x，于是我们便构造出最短路图。  我们发现，当两人所在起点到1号点最短距离不同时必然不会发生堵塞，于是我们按照最短距离长度将人分成若干组分别计算最多到达人数。  对于同组内的人，在最短路图的基础上添加一个源点，汇点为1，再添加从源点指向该组内的人的起点的边，此时最短路图边权已经毫无意义，所有边流量都是1，该图的最大流便是该组人最多可以到达1号点的人数。  官方数据真是丧心病狂，还卡网络流，而本题人数k很少，且边流量为0或1，因此可以直接找增广路做网络流，至多增广一次且每次增广时间均是O(m)的。 | 时间：O(nlogm+km)  空间：O(n+m) |
| 2013D | Factors | 设f(k)表示k分解质因数后质因子排列方案总数，给定n，问当f(k)=n时k的最小值是多少，保证1<k<2^63。  1<=n<2^63。 | 数论+搜索  设k=p1^e1\*p2^e2\*…\*pm^em，其中p1到pm是k的质因子，则f(k)=(e1+e2+…+em)!/(e1!+e2!+…+em!)，因为求的是最小值，则有p1=2，p2=3，p3=5，…且e1>=e2>=e3>=…>=em。  因此我们可以先枚举满足条件的所有k值，再根据输入的n从预处理出的数中找出最优的。  设有case组数据，满足要求的k值有tot1个，tot1= 43606，而f(k)满足1<=f(k)<2^63，则实际有tot2个符合，tot2=19274。 | 时间：O(tot1+tot2\*case)  空间：O(tot2) |
| 2013E | Harvard | 有k个变量，n个内存库，每个内存库有m个的空间，需要将k个变量分配到内存库中，调用1号内存库内变量不用改指针，调用其他内存库内变量需要先让指针指向那个内存库（若已经指向了就不必改）再调用，初始时指针指向空。给出使用变量的程序（有循环语句<Rn…E>和使用变量语句Vi），问在最优分配变量的情况下，最少的操作次数（使用变量和修改指针都算1次）。  1<=k,n,m<=13，k<=n\*m，变量使用次数<=10^12，程序有len句语句，len<=1000。 | 搜索  首先我们枚举哪些变量放到1号内存库中，贪心思想告诉我们肯定是放尽量多的变量到1号库。并预处理出变量总使用次数。  接着我们根据给定的程序，在忽略放入1号内存库的变量后，计算出所有f[i][j]表示使用变量i，它之前是变量j的次数，这里我们用递归处理程序中的循环即可。  接着我们再枚举剩余所有变量分配到n-1个内存库的情况，这里有一个强剪枝：搜索到某一步时，当变量最少的两个内存库变量数之和≤m时，它们合起来一定不会比分开劣，我们先计算出不存在上述情况时至少还需多少变量，若剩余变量数不够就剪枝。然后根据之前算出的f数组和变量分配方案，我们可以确定指针移动次数，加上变量总使用次数后，求出最少值便是答案。  可以证明搜索量不会超过贝尔数，其中B13=27644437。 | 时间：O(Bk\*k2+C(k,m)\*len)  空间：O(len+k2) |
| 2013F | 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，能量是k个电池的能量的最小值，机器的能量是芯片能量之差。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的最大值最小，问最小的值是多少。  2nk <= 10^6。 | 贪心  先将2nk个电池从小到大排序，设电池为p1,p2…,p2nk，则一个机器的两个芯片能量之差最小时选择的必然是相邻的两个电池作为最小值。再二分答案判断最小值能否小于等于它。我们发现如果选择了pi,pi+1，则还需要从pi+2到p2nk之间选择2k-2个电池，因此使用pi到p2nk至多满足(2nk-i+1)/(2k)个机器。由于i越大限制越紧，我们从大到小贪心选择，能选就选，如果最后选择达到n说明可行否则不可行。  令s=2nk，t=max(pi－pi-1)。 | 时间：O(slogs+slogt)  空间：O(s) |
| 2013H | Матрёшка | 有一排n个外表相同俄罗斯套娃，每个套娃有一个整数ai表示它是第几小的套娃，只有当一个套娃集包含恰好1-m各一个套娃才是完整的。你每次可以将一个玩偶或套娃集放入更大的玩偶中，且只能合并相邻的套娃，同时合并的玩偶不能再拆开来。如合并[1,2,6]与[4]，你需要将大小为4和6的两个玩偶拆开；合并[1,2,5]与[3,4]代价为3。求将这n个套娃合并成若干个完整套娃集的最少代价。 | DP  设f[i]表示前i个套娃合并成若干完整的套娃集的最少代价，g[i][j]表示第i-j个套娃合并成一个的最少代价，min[i][j]表示i-j中最小的套娃，max[i][j]表示i-j最大的套娃。  当i-j内有重复大小的套娃时g[i][j]=inf，否则g[i][j]=min{g[i][k]+g[k+1][j]+s}，其中s为最后一次合并的代价。而我们发现若某个大小的套娃比另外那个最小的小，则其与其内部所有套娃就不必拆开。  设t为i-j内最小套娃位置，k<t时让k从大到小循环，则min[i][j]单调不降且不必拆开的套娃必然在k+1-j内，不需要拆开的套娃数也满足单调不降；k>=t时让k从小到大循环，可得类似结论。因此我们可以均摊O(n)求出所有s。  而对于f的求解，根据f[i]=min{f[j]+g[j+1][i]}即可算出，f[n]便是答案。 | 时间：O(n3)  空间：O(n2) |
| 2013J | Pollution Solution | 求一个顶点数为n的多边形与一个半径为r圆心(0,0)的圆在x轴上方面积的交。  3<=n<=100，精度要求3位小数。 | 计算几何  由于精度要求不高，我们可以直接套用辛普森算法，在计算最终的图形在x=k上的长度时，我们直接算出多边形在其该直线上的点，点数必定是偶数且排序后第2i-1到2i个点间的部分在多边形内，可以简少代码量。 | 时间：O(nr/eps)  空间：O(n) |