IOI2014中国国家集训队第一次作业第二部分试题泛做表格

姓名：钟皓曦

试题编号

1998 B

题目名称

Flight Planning

题目大意

给出每一段的风速、飞机的单位时间耗油、不同的飞行高度和攀爬高度给飞机带来的油耗，问飞机的最少油耗已经达到该油耗的最小字典序的方案。

算法讨论

我们用$f[i][j]$表示第i段的飞行高度为j飞完剩下的所有路程的最小油耗。（这样表示是为了方便处理最小字典序的方案）那么转移就是枚举当前段和下一段的飞行高度算出代价即可。

时空复杂度

时间复杂度$O(N^3)$

空间复杂度$O(N^2)$

试题编号

1998 C

题目名称

Lead or Gold

题目大意

给出每种物品的三种配方混合比，问能否做出某个特定的三种配方比例的物品。

算法讨论

假设给出的物体三种配方比例是$x:y:z$，我们首先将x、y、z调整为$x+y+z=1$。那么这个时候只要有x和y就能够确定z 了。所以任意一种物品都可以只用x和y来表示。那么我们接下来就只需要考虑二维的情况了。我们将每种物品表示为平面上的一个点(x,y)。那么一种比例的物品能够被混合出来的条件就是它所表示出来的点在可用物品所组成的凸包内部。所以我们将点全部表示出来之后做出凸包之后检验最后的点是否在凸包内部即可。

时空复杂度

时间复杂度$O(N^3)$

空间复杂度$O(N)$

试题编号

1998 D

题目名称

Page Selection by Keyword Matching

题目大意

给出每个网站的关键字，每次询问询问与给出关键字关联度前五大的网页编号。

算法讨论

直接模拟算出每个网站的关联度然后排序即可。注意大小写是没有区分的。

时空复杂度

时间复杂度$O(NM)$

空间复杂度$O(NM)$

试题编号

1998 E

题目名称

Petri Net Simulation

题目大意

有N个仓库和M种转存方式。每种转存方式会从一些仓库中取出一定数量的物品，然后在另外某些仓库中放入某些物品。一个转存方式是合法的当且仅当能够拿出足够数量的物品。你每次可以从合法的转存方式中选一种出来进行转存。问能否成功进行T次转存，数据保证无论你每次怎样选取转存方式最后得到的结果相同。

算法讨论

直接模拟即可。每次枚举进行哪一种转存方式，能进行就直接进行。:D

时空复杂度

时间复杂度$O(NMT)$

空间复杂度$O(NM)$

试题编号

1998 G

题目名称

Spatial Structures

题目大意

给出一个矩阵或者一个矩阵的黑点象限四分树，求该矩阵的黑点象限四分树或者该矩阵。

算法讨论

直接模拟即可。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N^2)$

试题编号

1999 A

题目名称

Bee Breeding

题目大意

一张无限大的六边形网格图，每次可以从一个六边形走到周围的六个六边形，问某两个六边形之间的最短距离。

算法讨论

我们将给出的1号点的坐标定为$(0,0)$，将从左上到右下的线作为Y轴，将竖线作为X轴建立坐标系。那么我们会发现从一个点能走出去的六个方向的(deltax,deltay)为$(1,0)$、$(1,-1)$、$(0,-1)$、$(-1,0)$、$(-1,1)$和$(0,1)$。那么接下来可以算出询问的两点的坐标后进行bfs 求解即可。(也可以推公式，因为我们可以发现如果将现有坐标转移到方格坐标上后从一个方格向周围的八个方向仅有两个方向不能走，所以可以按象限进行分情况讨论)

时空复杂度

时间复杂度$O(1)$

空间复杂度$O(1)$

试题编号

1999 C

题目名称

A Dicey Problem

题目大意

给出一张地图和一开始一个骰子的状态（位置和方向）。每次你可以把骰子向某个方向进行翻转移动，要求移动到的地方的数值与骰子的顶面数字相等（或者数值为–1）。要求找出一条从起点出发再回到起点的移动路径。

算法讨论

由于即使骰子到达同一个地方但是各个数字的朝向也会不一样，所以我们用一个思维状态表示走到某个地方且此时的顶面数字和正前方数字各是多少（只需要这两个数字即可确定骰子的状态）。在这个基础上在做bfs即可。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N^2)$

试题编号

1999 D

题目名称

The Fortified Forest

题目大意

有N棵树，要求砍掉最少代价的树，使得获得的木材能够将剩下的树围住。

算法讨论

我们枚举我们需要砍掉哪些树，那么此时能够获得的木材以及花费的代价都确定了。对于剩下没有被砍掉的树，我们要将它们围住的话，一定是围住它们所形成的凸包。所以我们对于剩下的树组成的图形求一个凸包算出周长即可判断是否满足条件了。

时空复杂度

时间复杂度$O(2^N * NlogN)$

空间复杂度$O(N)$

试题编号

1999 E

题目名称

Trade on Verweggistan

题目大意

有N堆物体，每堆的每个物体有一个成本，每个物体可以带来10的收益。每次你可以从一堆中选取最上面的几个物体拿走，求最后的最大收益以及能达到这个收益需要拿走多少个物体。

算法讨论

最简单的背包，用$f[i][j]$表示从前i堆物体中拿走了j个物体的最大收益，转移直接枚举从当前堆顶拿走多少物体即可。

时空复杂度

时间复杂度$O(N^2M^2)$

空间复杂度$O(NM)$

试题编号

1999 H

题目名称

flooded!

题目大意

有一个地方会降下总量为v的雨。先给出每个地方的高度，求最后的水位高度以及有多少地方被水淹没。

算法讨论

注意到积水一定是从最低的地方开始形成，所以我们首先将所有的高度排序，然后从较低的高度向较高的高度进行扫描。每次扫描到下一个高度时先判断当前剩余雨量是否能够达到下一高度，如果不能直接退出，否则继续向后进行扫描即可。

时空复杂度

时间复杂度$O(N^2 logN^2)$

空间复杂度$O(N^2)$

试题编号

2000 A

题目名称

Abbott’s Revenge

题目大意

给出一张网格图，对于每个点你都有四种可以走到它的方向。对于每种走到它的方向，会限制你能够走出去的方向。现在给出该网格图，求从给定起点到给定终点的最短距离。（一开始会强制你从起点向着某个方向走一步）

算法讨论

很基础的BFS，只需要注意一开始会被强制向某一个方向走一步处理掉环的情况即可。

时空复杂度

时间复杂度$O(NM)$

空间复杂度$O(NM)$

试题编号

2000 B

题目名称

According to Bartjens

题目大意

给出一个被去掉符号的算式，要求填上+、−、∗三种符号之后使得原式的值为2000。

算法讨论

直接搜索即可。:D

时空复杂度

时间复杂度$O(4^N)$

空间复杂度$O(N)$

试题编号

2000 C

题目名称

Cutting Chains

题目大意

有一些环套在了一起，你需要打开某些环然后将它们进行移动后关闭，使得最后所有环形成一条链。问最少需要打开的环的数量。

算法讨论

我们首先枚举打开哪些环，然后将与之相连的所有环全部分开。由于接下来我们不会再打开任何的新环，所以此时相连接的环都必须组成一条链，如果不是链的话无论怎么组合最后的结果也不可能是链。假设最后总共形成了x个联通块，我们打开了y个环，那么最后能够形成链的条件是$x <= y * 2 + 1$。（即用一个打开的环去连接两个块）所以我们枚举打开哪些环后，判断是否全部为链以及是否满足该不等式即可解决问题。

时空复杂度

时间复杂度$O(2^N * N^2)$

空间复杂度$O(N^2)$

试题编号

2000 E

题目名称

Internet Bandwidth

题目大意

求一个无向图的最大流。

算法讨论

直接做网络流即可。:D

时空复杂度

时间复杂度$O(N^2 * M)$

空间复杂度$O(N + M)$

试题编号

2000 F

题目名称

Page Hopping

题目大意

求所有点对之间的平均最短路。

算法讨论

因为有$N <= 100$，所以直接做floyd即可。:D

时空复杂度

时间复杂度$O(N^3)$

空间复杂度$O(N^2)$

试题编号

2001 A

题目名称

Airport Configuration

题目大意

给出城市与城市之间的人流量以及很多种排布机场的方案。将每种方案的代价排序后输出。（一种方案的代价为所有人需要移动的距离总和）

算法讨论

直接模拟算出每种方案的代价之后排序输出即可。

时空复杂度

时间复杂度$O(KN^2 + KlogK)$

空间复杂度$O(N^2)$

试题编号

2001 B

题目名称

Say Cheese

题目大意

一个无限大的实体空间中有N个球形部分为空（球形部分可能有重叠），你在空的空间里的移动速度为无限大，在实体空间中移动的速度为每秒1个单位。问从给定起点移动到给定终点的最短时间。

算法讨论

给定的起点和终点我们可以将其看做两个半径为0的球体，这样我们就统一了所有的部分。考虑从一个球体球心走到另一个球体球心的最短时间，如果这两个球相交的话那么两个球心只用0的时间就可以到达，否则则是它们圆心的距离减去它们的半径之和（也就是在实体空间内会移动的距离）。做出了所有球之间的距离之后，直接使用最短路的算法算出答案即可。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N)$

试题编号

2001 F

题目名称

A Major Problem

题目大意

将给出的某个大调音阶的音符转换为另一个大调音阶的对应音符。

算法讨论

直接模拟即可。

时空复杂度

时间复杂度$O(1)$

空间复杂度$O(1)$

试题编号

2001 I

题目名称

A Vexing Problem

题目大意

给出一张游戏地图，每次可以将一个位置上的方格向左或向右移到一个没有其他物品的位置上。两个相同的方块挨在一起就会消除，问最少消除步数以及方案。数据保证最后的步数不超过11步。

算法讨论

基本的思路就是搜索，搜索的过程中需要判重，以及加上两个剪枝。第一个剪枝是如果某种颜色只剩一个直接退出，另外一个是如果某种颜色如果剩下两个，那么它们横向的距离是必须靠自己的移动得到的，那么就可以算出一个最小的可能步数。用这两个剪枝即可过这道题了。

时空复杂度

时间复杂度$O(11^{nm})$

空间复杂度$O(nm * 11^{nm})$

试题编号

2002 A

题目名称

Ballons in a Box

题目大意

有边界的立体空间中有N个可以放置气球的点。每放置一个气球这个气球就会膨胀至不能再膨胀。问最后剩下的没被气球占据的空间最少是多少。

算法讨论

直接爆搜。:D

时空复杂度

时间复杂度$O(N * N!)$

空间复杂度$O(N)$

试题编号

2002 C

题目名称

Crossing the Desert

题目大意

给出一个平面坐标，你需要从起点走到终点。你有一个能携带的物品容量上限，你可以在起点无限购买食物和水或者在其他点得到水或者储存食物。现在问从起点走到终点至少需要多少食物。

算法讨论

我们用$dist[i]$表示从i点走到终点至少还需要购买多少食物。考虑如何转移$dist$。显然我们有$dist[n] = 0$，那么考虑从某个点走到这个点有两种情况：1、直接抵达。2、通过食物储存抵达。对于第一种情况，我们可以直接算出代价。对于第二种情况，我们可以算出每单位距离需要的食物储存量，那么通过这个值我们可以得出通过反复的食物储存抵达当前点的代价。有了以上两个信息，我们就可以用最短路来转移我们的$dist$值。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N)$

试题编号

2002 E

题目名称

Island Hopping

题目大意

给出平面上N个点，求出最小生成树，每个点的代价为其权值乘以它到1号节点路径上的最大边权。求总价值。

算法讨论

直接求出最小生成树后从每个点向根节点走一遍看路径上的最大值是多少即可。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N^2)$

试题编号

2002 H

题目名称

Silly Sort

题目大意

每次可以交换序列中的两个数，代价是两个数的和。求最小的代价使得序列有序。

算法讨论

考虑对于每一个置换群，我们有两种操作方式：1、在该置换群内部完成交换操作。2、从置换群外部找一个数辅助完成操作。第一种操作一定是每次用最小的数与其他数完成置换，第二种操作一定是用外部最小的数完成置换。那么对于每个置换群我们都这样检验一次就可以得出答案了。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N^2)$

试题编号

2003 A

题目名称

Building Bridges

题目大意

在一张网格图上有一些区域，你需要沿着网格线修建道路，用最短长度的道路连接最多的区域。

算法讨论

首先我们处理出哪些格子属于哪些区域。那么接下来的问题解决就是我们要如何修建道路。实际上我们如果处理出所有的道路之后就是一个最小生成树的问题了，那么处理出可以修建的道路的方法就只需要从每个交点向四个方向暴力出会遇到哪个区域即可。

时空复杂度

时间复杂度$O(N^3)$

空间复杂度$O(N^2)$

试题编号

2003 B

题目名称

Light Bulbs

题目大意

给出N盏灯的初始状态和结束状态，每个灯上的开关会改变自己和相邻两个的灯的状态（如果是边界上就只有一个相邻的）。问最少多少次由起始状态改变为终止状态。

算法讨论

可以发现当我们确定了是否对一个开关操作后我们就可以知道接下来究竟是否对某个开关进行操作。所以我们直接枚举两种情况，算出每一种的答案即可。注意高精度。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N)$

试题编号

2003 D

题目名称

Eurodiffusion

题目大意

每个城市会将自己的硬币分一些给周边城市，问什么时候所有城市都有所有种类的硬币。

算法讨论

直接模拟即可。

时空复杂度

时间复杂度$O(N^2 * T)$

空间复杂度$O(N^2)$

试题编号

2003 H

题目名称

A Spy in the Metro

题目大意

你需要从起点走到终点且必须使用指定数量的时间。每次你可以换乘反方向的车或者在车站等待，问完成要求所需要的等待时间。

算法讨论

我们用$dist[i][j]$表示到达i号站台且当前用时为j所需要的最少的等待时间。那么我们可以用最短路来进行转移，每次转移就只需要考虑等待、向前和向后三种情况即可。

时空复杂度

时间复杂度$O(N^4)$

空间复杂度$O(N^2)$

试题编号

2003 I

题目名称

The Solar System

题目大意

求一个星球在运动一定时间后的坐标。

算法讨论

根据开普勒第一定律，行星与焦点的连线扫过的面积在单位时间内是相同的。由于我们知道行星的运动时间和运动周期，所以我们是可以算出行星扫过的面积的。我们只考虑此时行星在x轴上方的情况（在下方是一致的）。此时随着横坐标的增长，行星与焦点连线扫过的面积也是增长的。所以我们可以二分横坐标的位置，每次对所需要计算的部分的面积进行定积分得到答案即可。需要long double来提高精度。

时空复杂度

时间复杂度$O(1)$

空间复杂度$O(1)$

试题编号

2003 J

题目名称

Toll

题目大意

给出一张图，某些点是城市而某些点是乡村。走到城市需要缴纳与携带货物成正比的税，而走到乡村需要缴纳定量的税。现在需要运送一些货物，问最少需要缴纳的税费。（税费即货物）

算法讨论

可以看出如果我们携带某个货物量能够实现成功运输的话，那么更多的货物量也一定能够成功运输。也就是说，能否成功运输与携带货物量是有单调的关系的。所以我们可以二分初始携带的货物量，每次做SPFA判断能否成功运输即可。

时空复杂度

时间复杂度$O(N^3logN)$

空间复杂度$O(N)$

试题编号

2004 H

题目名称

Tree-Lined Streets

题目大意

给出平面上N条线段，你需要在线段上标记尽量多的点。对于同一条线段上被标记的点，它们之间的距离不能小于50，它们到该条线段与其他线段的交点的距离不能小于25。问最多能标记多少个点。

算法讨论

两两枚举所有的线段组合，求出它们的所有交点。然后由于线段与线段之间是不影响的，所以我们可以只考虑一条线段的情况。一条线段上的交点会把该条线段分成很多段，中间的段的长度会被裁掉50，而两边的线段长度只会被裁掉25。算出了实际有用的长度之后，那么对于单独的有用长度为L一条线段的答案就是$\lfloor \frac{L}{50} \rfloor + 1$。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N)$

试题编号

2005 A

题目名称

Eyeball Benders

题目大意

给出两张图，每张图都是由一些垂直于坐标轴的线段组成的图。现在询问是否能够将第一张图缩小一定倍数（缩小之后要保证最短线段的长度大于等于0.005），然后进行平移操作，使得它变为第二张图中的一部分。你需要保证第一张图中的某个线段端点在经过一系列操作后是第二张图中某条线段的端点。数据保证第一张图内部同向的线段不会重叠，不同向的线段至少有一个交点。

算法讨论

考虑该问题，我们首先需要确定缩小的比例和平移的向量。第一点，由于我们需要保证第一张图的一个端点能够映射到第二张图上的某个端点上去，所以我们可以首先枚举第一张图中的某个点映射到第二张图上的哪个点上去。有了这组映射关系我们就可以确定第一张图的平移量（平移和缩小两种操作互不影响）。接下来我们还需要一组映射关系才能确定缩小的比例，注意到数据保证了不同向的线段至少有一个交点，那么我们可以任取一个交点，然后在第二张图中枚举第一张图中的这个交点映射到第二张图中的哪个交点（因为经过变换之后交点依旧是交点），有了这组映射关系我们就可以确定了缩小量。那么确定了平移的向量和缩小的向量后，我们便可以将第一张图经过变换之后的图表示出来。表示出来后，我们就可以确定新图的上下左右边界，有了边界我们就可以将第二张图中属于这一部分的线段抠出来。接下来的判重问题就是一个简单的排序然后扫一遍就能解决的问题了。

时空复杂度

时间复杂度$O(N^4 * \log(N))$

空间复杂度$O(N)$

试题编号

2005 C

题目名称

The Traveling Judges Problem

题目大意

有N个城市和M个人，给出M个人的初始位置。这M个人都需要走到一个固定的终点去，最后的总代价为所有人所经过的路径的并的长度。求使得这个长度最短的方案。

算法讨论

我们用$f[i][j]$表示集合为j的人都到达i号城市的最小花费。那么对于一个特定的i和j，首先我们枚举j的子集k，那么就有$f[i][j] = min(f[i][k] + f[i][j \otimes k])$。对于特定的j做出了所有的$i$的答案之后，我们可以在同一层内做spfa来做对第一维的转移。这样我们就成功的做出了这个DP。

时空复杂度

时间复杂度$O(N * 2^M * (3^M + N^3))$

空间复杂度$O(N * 2^M)$

试题编号

2005 E

题目名称

Lots of Sunlight

题目大意

给出N幢楼的高度和宽度以及所在位置。询问某些楼房的某些层被阳关照射的时间。

算法讨论

在处理某个询问的答案时，我们只需要知道左边的楼房对它造成的最大影响和右边的楼房对它造成的最大影响就行了（所谓的影响，就是因为这幢楼的存在，会使得其被照射时间的改变量，也就是该幢楼的顶部与询问点所成的角度）。所以我们直接枚举左边和右边的楼房计算影响即可。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N)$

试题编号

2005 H

题目名称

The Great Wall Game

题目大意

$N*N$的棋盘上有N个棋子，求最少的步数将这N颗棋子移动到一条直线上。

算法讨论

我们考虑实际上可能的情况只有三种：1、竖线。2、横线。3、斜线。对于前两种情况（比如竖线），最后所有的棋子的y坐标的值都要相等，所以我们可以直接算出横向的移动量。对于纵向的移动量，显然让x尽量小的棋子去x更小的地方是更优的，所以我们可以贪心得出答案。对于第三种情况，我们可以把每个棋子和它们所需要到的位置建立一张二分图，边的边权为两点距离（对于前两种情况也可以这么做），做一次费用流即可得出答案。

时空复杂度

时间复杂度$O(N^2+N^3)$

空间复杂度$O(N^2)$

试题编号

2005 I

题目名称

Workshops

题目大意

有N个会议在同一时刻开始，每个会议有一定的人数和持续时间。有M个可以开会的地方，每个地方有最多能够容纳的人数和最多能提供的时间。现在问最多能使多少人成功参加会议。

算法讨论

由于我们最后需要使最多的人参加会议，所以我们贪心的从人数最多的会议开始选取。考虑当前会议如果有多个能够满足它的地点，那么我们只需要选取能够满足的当中能够提供的时间最少的即可（因为我们是按人数贪心，当前能满足该会议的场地的在人数上也一定能满足以后的会议）。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N)$

试题编号

2005 J

题目名称

Zones

题目大意

给出N个集合的大小以及其中某些集合交的大小（没有给出的交集的大小都为0）。现在从中选K个集合，问它们并集的大小最大是多少。

算法讨论

直接枚举选哪K个集合，然后用容斥原理算出这些集合的并集大小即可。:D

时空复杂度

时间复杂度$O(\binom{N}{K} * M)$或者$O(2^N * M)$

空间复杂度$O(1)$或者$O(2^N * M)$

试题编号

2006 A

题目名称

Low Cost Air Travel

题目大意

给出所有可以乘坐的航班，问实现某条旅行线路的最小代价。

算法讨论

首先用floyd预处理出从某一个航班的某个位置到达另外一个航班的某个位置的代价，在每次询问时再做最短路即可。

时空复杂度

时间复杂度$O(1000NT^3 + 10NI * 100 * NT^2)$

空间复杂度$O(10NT^2)$

试题编号

2006 E

题目名称

Bit Compressor

题目大意

给出原串有多少个1和多少个0,以及压缩后的串。压缩的方法为将连续的超过两个的1变为它们个数的二进制表示。求原串。

算法讨论

由于总长度不超过40，所以直接搜索即可。:D

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N)$

试题编号

2006 I

题目名称

Degrees of Separation

题目大意

给出一张图，就两点之间最短路的最大值。

算法讨论

枚举其中一点，每次做最短路求出最大值即可。

时空复杂度

时间复杂度$O(N^3)$

空间复杂度$O(N^2)$

试题编号

2006 J

题目名称

Routing

题目大意

从起点走到终点再走回来，问最少需要经过多少个不同的点。

算法讨论

首先我们可以将从起点走到终点再走回来看作是从起点走两条路径到终点，只不过其中一条路径是反图。那么我们可以用$f[i][j]$表示两条路径分别走到了i和j号点时的最小花费。考虑转移实际上就是枚举从i走到某个点或者是从j走到某个点。但是由于边权不为1，所以我们需要使用spfa但这样会TLE。注意到所有的边权都是小于等于1的，所以我们可以使用多个队列bfs降低复杂度。

时空复杂度

时间复杂度$O(N^3)$

空间复杂度$O(N^2)$

试题编号

2007 A

题目名称

Consanguine Calculations

题目大意

给出基因与血型的关系表，告诉父亲、母亲、儿子三个人中两个的血型问剩下一个人可能存在的合法血型。

算法讨论

直接暴力搜索即可。:D

时空复杂度

时间复杂度$O(6^3)$

空间复杂度$O(1)$

试题编号

2007 I

题目名称

Water Tanks

题目大意

给出一个储水设施，除了第一个出水管剩下的出水管的顶部都是封闭的。连接相邻两个储水设置的管道高度是单调递增的。问注满水后能够存下多少容量的水。

算法讨论

由于连接两个储水设施的管道高度是单调递增的，所以后面的注水过程是与前面的注水过程是无关的，所以我们可以从左至右一个一个储水设施的做。对于每一个储水设施，我们可以将其的注水过程分为三个部分：1、达到前一个管道的高度、2、到达下一个管道的高度。3、注满水。所以我们可以按照这个过程算出每一个部分能不能够到达就能够得到当前储水设施的储水量了。

时空复杂度

时间复杂度$O(N)$

空间复杂度$O(N)$

试题编号

2007 J

题目名称

Tunnels

题目大意

给出一张图，你可以在任意时刻删除一些边使得某个人不能到达出口，求最坏情况下需要删除的边的数目的最小值。

算法讨论

首先我们可以做出当那个人在某个点时我们不希望他进行移动所需要删除的边的数目。我们考虑如果一开始那个人选择了一个代价很大的点，那么我们可以通过封住某些路使他移动到并不需要那么多代价的点上去。所以我们可以在这里用最短路来松弛每个点的权值，两个点之间的权值再做一次网络流即可得到。

时空复杂度

时间复杂度$O(N^3M)$

空间复杂度$O(N+M)$

试题编号

2008 A

题目名称

Air Conditioning Machinery

题目大意

给出一个立方体和一个特定的形状，你需要用不超过六个特定的形状从起点拼接出一条到终点的路，问最少需要多少个立方体。

算法讨论

直接爆搜。:D

时空复杂度

时间复杂度$O(8^6)$

空间复杂度$O(N^3)$

试题编号

2008 B

题目名称

Always an Integer

题目大意

判断一个多项式在x的值取不同的正整数时是否恒为整数。

算法讨论

由于最高次项的次数不会超过100，所以我们只需要将x赋值为连续的100个整数带入检验，就一定能够知道该多项式是否恒为整数。

时空复杂度

时间复杂度$O(N)$

空间复杂度$O(N)$

试题编号

2008 D

题目名称

The Hare and the Hounds

题目大意

给出一张方位图。定义猎狗和兔子的基本移动方式是走转角最小的路。兔子在某些点会随机选择移动方式，猎狗在这些点会执行检查过程直到检查到兔子走过某个地方。问兔子的猎狗移动的距离以及兔子移动的路径。

算法讨论

直接模拟。:D

时空复杂度

时间复杂度$O(360N)$

空间复杂度$O(360N)$

试题编号

2008 E

题目名称

Huffman Codes

题目大意

给出一个经过哈夫曼编码之后的序列，问可以构成该编码序列的原序列有多少种。

算法讨论

首先第一步我们想办法建出哈夫曼树。我们将我们得到的所有编码序列排序，那么此时最小的两个值一定是哈夫曼树上的某个结点的左右儿子。我们不断做这个过程，直到我们找到了哈夫曼树的根。处理出了哈夫曼树之后，我们观察到实际上可用的变量以及变量的最大值都不会很多，所以我们直接暴力搜索出合法的序列即可。

时空复杂度

时间复杂度$O(100^N)$

空间复杂度$O(N)$

试题编号

2008 F

题目名称

Glenbow Museum

题目大意

定义一个直角多边形的90°角为R，270°角为O，那么一个普通的矩形可以表示成为RRRR。现在询问长度为N的序列中有多少个不存在连续两个O并且最后是一个合法的直角多边形。

算法讨论

考虑最简单的情况就是$N = 4$。对于$N = 4$的时候，我们如果希望当中出现O，就只能将某个90°换成一个270°，也就是将一个R 替换为ROR。所以我们想要在原来的序列上得到一个更长的长度，我们唯一的方法就是将某个R替换为ROR。（所以，长度为奇数或者没有达到4的情况都无解）。最后的长度为N，也就是说最后的序列当中有$4 + \frac{N}{2}$个R。我们假设$x = 4 + \frac{N}{2}$，那么我们有我们最后的答案$ans = \frac{\binom{x}{4} * \binom{x-1}{4}}{24}$，于是我们可以在$O(1)$的时间内求出答案。

时空复杂度

时间复杂度$O(1)$

空间复杂度$O(1)$

试题编号

2008 G

题目名称

Net Loss

题目大意

给出一个多项式，求出一个与其之差的平方的在$[-1,1]$上定积分最小的的分为两段的线性变换函数。

算法讨论

题目给出了分段函数的分段点的横坐标，我们可以发现当分段点的y值确定之后，函数的左边和右边是没有联系的。只考虑左边我们会发现左边这一段的定积分的值是一个关于左端点的y值得一个二次函数，具有凸性，所以我们可以三分出当前状况下左边最好的y值以及函数值。右边同理。同理我们可以发现整个函数的函数值是关于断点y值的一个二次函数，所以确定断点的y值也可以进行三分。用一个三分套三分即可解决问题。

时空复杂度

时间复杂度$O(50*50*20*20)$

空间复杂度$O(1)$

试题编号

2008 H

题目名称

Painter

题目大意

给出平面上N个三角形，问是否有三角形相交。如果没有，问被包含次数最多的三角形的被包含次数。

算法讨论

我们首先考虑如何判断三角形是否相交。这个问题的本质其实是问平面上的很多条线段是否相交，我们可以将线段按照x的坐标进行排序然后用平衡树维护扫描线扫描到的东西。一旦两条线段在平衡树内的位置发生交换就说明出现了相交的情况。同理，在处理被包含次数时也可以直接使用扫描线加平衡树维护即可。

时空复杂度

时间复杂度$O(N \log N)$

空间复杂度$O(N)$

试题编号

2008 I

题目名称

Password Suspects

题目大意

问长度为N的字符串中有多少个是包含了给出的给定的M个串的，在答案小于一定值的时候输出方案。

算法讨论

我们对给出的M个串建立AC自动机，那么我们可以在AC自动机上进行DP。用$f[i][j][k]$表示我们当前的字符串长度为i走到了自动机上的j号结点且此时已经出现了的字符串的集合为k的方案数。那么转移就直接从每种状态枚举下一位填什么字母即可。对于输出方案，暴力搜索即可。

时空复杂度

时间复杂度$O(N * M^2 * 2^M * 26p)$

空间复杂度$O(N * M^2 * 2^M)$

试题编号

2008 J

题目名称

The Sky is the Limit

题目大意

给定底边都在x轴上的多个等腰三角形，现在问所有三角形的并的周长是多少（并之后询问周长时不计算底边长度）。

算法讨论

由于三角形的两条腰边都是线段，都是属于线性的变换。所以我们需要计算的答案可以拆分为一条一条的线段，而可能出现拐点的地方就只有在原来各条线段对应的交点处。在相邻的交点与交点之间，一定是由一条线段组成的答案。所以我们可以预处理出所有交点的位置然后将其排序，算出每个交点处函数所对应的值，然后在通过相邻两个交点的函数值算出这一段的答案即可。

时空复杂度

时间复杂度$O(N^3)$

空间复杂度$O(N)$

试题编号

2009 A

题目名称

A Careful Approach

题目大意

给出N架飞机的可降落时间，问可能的最长的两架飞机的降落时间差。

算法讨论

首先可以很容易的看出如果某个降落时间差是可行的，那么更短的时间差也一定是可行的。所以我们可以二分答案。又由于$N <= 8$，所以我们每次可以暴力搜索出所有可能的降落顺序然后从左至右贪心放置即可。

时空复杂度

时间复杂度$O(N * N!)$

空间复杂度$O(N)$

试题编号

2009 G

题目名称

House of Cards

题目大意

给出N张牌，每张牌有花色和点数。将最开始的八张牌摆在桌上与桌面形成四个三角形。以后每次两人轮流放牌，可以执行的操作有：1、屯牌。（至多一张）。2、将一张卡片横放形成一个倒三角。3、将两张卡片竖放形成一个竖三角。每个三角形的权值是三张牌的点数总和，权值会加给三张牌中花色较多的一边。最后剩下的牌的点数会加入对应的人的分数中。现在双方都采取最优策略，问两方的分差。

算法讨论

很容易看出该问题是个搜索问题，但直接的爆搜的复杂度是不能接受的。由于该问题是一个双方较量的问题，所以我们可以直接使用alpha-beta剪枝。使用了alpha-beta剪枝之后会被搜索的状态就大大减少了，可以顺利通过该题。

时空复杂度

时间复杂度$O(N!)$

空间复杂度$O(N)$

试题编号

2009 I

题目名称

Struts and Springs

题目大意

有N个矩形，每个矩形由六个弹簧或者支杆固定。每个矩形周围的四个固定轴一定都连到另外一个矩形的四条边上。现在不断改变最外层矩形的大小，求内部每个矩形的大小。

算法讨论

N只有500，暴力做出所有矩形的嵌套情况以后每次暴力修改每个矩形即可。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N)$

试题编号

2009 J

题目名称

The Ministers' Major Mess

题目大意

每个人会对不超过4个议案进行投票，要求最后对于每个人投票的议案的结果有超过一半是与投票人的看法是一致的。问是否可行，以及每个议案是必须通过或者否定或者两者皆可。

算法讨论

由于每个人投票的议案不会超过四个，我们设为k。那么对于$k \leq 2$的人，他投票的所有议案都必须符合他的要求；对于$k > 2$的人，至多只会有一个不满足他要求的议案。并且每个议案有且仅有通过与否定两种矛盾的状态，那么这就可以看做是一个基本的2-SAT模型。对于$k \leq 2$的人会确定某些必须被选取的点。对于$k > 2$的人，可以根据只要有一个议案不满足要求那么剩下的议案都必须满足要求来建立矛盾关系。建出了2-SAT模型后，再检验一遍即可了。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N)$

试题编号

2010 B

题目名称

Barcodes

题目大意

定义条形码为由一些区域从左至右排列而成的图案。区域之间的颜色深浅交替（该条件无用），并且每个区域有一个宽度。可能的宽度总共有两种，较宽的那种的宽度是窄的宽度的两倍。条形码中相邻的五个区域可以被解码为一个字符，对应的解码方法在原题的表格中给出。每两个相邻字符所表示出来的10个区域之间会用一个窄的区域进行分割。一个字符串在被编码为条形码时，会在该字符串尾部添加两个检验字符（检验字符的计算方式题目给出），在添加完检验字符后会在字符串头部和尾部加入开始和结束标记后再按编码规则编码为条形码。现给出条形码每个区域的宽度，询问该条形码是否合法。给出不合法的信息或者编码前的字符串。在读入宽度时你所得到的数据会与原数据有不超过5%的误差。输入的宽度序列有可能是逆序的。

算法讨论

根据题目当中给出的字符的编码方式我们可以发现，无论我们读入的顺序是顺序的还是逆序的，如果该序列是一个合法的序列，那么第一个区域的宽度一定是窄区域的宽度。那么我们可以直接将第一个区域设置为窄区域。即使第一块区域的宽度因误差导致其数值是真实宽度的95%，那么最宽的窄的宽度也不会超过该宽度的21/19倍。所以我们可以将大于该临界值的全部设为较宽的区域，将剩下的全部设为较窄的区域。经过这样的处理之后我们就成功的消去了误差，剩下的问题就是简单的模拟了。

另外该题有很多细节，在这里罗列一下：

1、某种宽度的误差超过了5%。

2、起始结尾不为开始结束标记。

3、中间部分出现了开始结束标记。

4、较宽部分的宽度不为较窄部分宽度的两倍。

5、两个检验字符的值不正确。

6、序列有可能是逆序的。

注意到了这些细节，加之前面部分的误差处理后，就能够成功的解决这道题了。

时空复杂度

时间复杂度$O(N)$

空间复杂度$O(N)$

试题编号

2010 D

题目名称

Castles

题目大意

攻占编号为i的城市至少需要Ai名士兵，会牺牲Bi名士兵，需要派Ci名士兵留守。总共有N座城市，它们之间形成一棵树。你可以随意选取你第一个攻占的城市。在整个攻占行动中，每条边至多被经过两次，每次只能从已经攻占的城市去攻占未被攻占的城市。问最少需要的士兵总数。

算法讨论

枚举我们第一个攻占哪个城市，那么接下来需要解决的是以这个点为根最少需要多少士兵才能攻占完所有的城市。考虑在处理以i为根的子树时的答案，假设j和k都是i的儿子，并且我们已经处理出了攻占以j和k为根的子树最少需要的士兵数need和攻占完成后会减少的士兵数delta。考虑如果只有j和k两个儿子的话，我们需要决定的是先攻占哪一棵子树。先攻占j的代价为max(need[j],delta[j]+use[k])，先攻占k的代价为max(need[k],delta[k]+use[j])，我们只需要比较这两个值即可确定攻占顺序。当有多个儿子时，我们只需要以这个作为关键字排序即可确定这些子树的攻占顺序。接下来我们只需要扫一遍即可做出当前子树的答案。

时空复杂度

时间复杂度$O(N^2 * \log N)$

空间复杂度$O(N)$

试题编号

2010 F

题目名称

Contour Mapping

题目大意

给出一个由三角形拼成的六边形网格图，每个结点有个高度。相同高度的地方形成等高线。现在询问所有等高线高度为H的倍数的等高线长度。

算法讨论

考虑三角形与三角形之间是没有影响的，所以我们可以单独算出每个三角形中等高线的数值。在同一个三角形中，每条等高线的长度形成一个等差数列，所以我们可以求出最长的等高线长度和最短的等高线长度以及等高线数量（计算等高线长度可使用正弦定理），那么我们就可以出该三角形内部的等高线长度和。接下来需要处理的是多个相同高度的点导致等高线的计算重复，这一步可以使用bfs处理出等高线边缘或者扫一遍判一条边是否在该区域内部即可。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N^2)$

试题编号

2010 G

题目名称

The Islands

题目大意

找出一条从出发点一直往东到达终点后在一直往西回到起点且每个城市只经过一次并且某两个特殊城市是出于两段不同的旅行中且总路程最小的旅行路线。

算法讨论

一次向西一次向东，实际上可以看做是两次向东且旅行路程中没有相同的点（起点终点除外）的两条路径。我们设$f[i][j]$表示第一条线路走到了i点第二条线路走到了j点的最小代价。由于我们需要经过所有的点，那么此时$1-max(i,j)$的所有点都应该已经被经过了。那么下一个需要走的点是$max(i,j)+1$。那么如果这个点不是两个特殊点中的一个的话，那么我们可以直接枚举是第一条线路走过来还是第二条线路走过来，否则去除其中之一的转移即可。（强制使第一个特殊点在第一条线路中，第二个特殊点在第二条线路中）

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N^2)$

试题编号

2010 I

题目名称

Robots on Ice

题目大意

给出一张方格图，其中有四个点是必须在指定时间到达的。问遍历整张图的方案数。（每个点只经过一次）

算法讨论

数据范围不是很大，但是直接的暴力搜索是过不了的，需要剪枝优化。最基础的剪枝即为提前到达了某个指定点或者到达了指定时间却不在指定点上。我们实际上还可以判断当前点到下一个指定点的这段路是否能在余下的时间内完成，如果不能这也显然是一种不合法的方案。由于每个点只能经过一次，所以一旦某个时刻我们将图分为了两个部分且两个部分内都还有没到达过的点，那么接下来我们无论如何也不可能完成对图的遍历了。有了以上的剪枝之后就可以顺利通过该题了。

时空复杂度

时间复杂度$O(4^{NM})$

空间复杂度$O(NM)$

试题编号

2010 J

题目名称

Sharing Chocolate

题目大意

给出一块长x宽y的巧克力，你需要把它分成N部分，每次分割只能沿着某块巧克力的某条竖线或者横线切开（都是整数位置）。现给出N个部分的大小，问能否成功分割。

算法讨论

用$f[i][j]$表示当剩下的巧克力的某一个长度为i能否成功分割成集合j。由于集合j的$size$是确定的，所以我们当前巧克力的另外一个长度一定为$\frac{size[j]}{i}$。确定了两条边的长度后，我们枚举j的子集，把剩下需要形成的部分分为两部分，然后再考虑是竖着进行分割还是横着进行分割（两种分割所需要的巧克力的长宽都可以直接计算）进行记忆化搜索即可。

时空复杂度

时间复杂度$O(X*(2^N+3^N))$

空间复杂度$O(X*2^N)$

试题编号

2010 K

题目名称

Paperweight

题目大意

给出一个由两个四棱锥拼接而成的立体图形，定义这个图形是稳定的当且仅当我们选定这个图形的底面之后将重心向任意方向移动之后不超过0.2个单位距离后该图形都不会移动。问所有能使该图形稳定的方案中某个特定点到底面距离的最大值。

算法讨论

由于总共的点数和面数都是常数，所以我们直接枚举我们所选取的底面。选取了底面之后，原问题转化为：将重心投影到当前底面后，投影出来的点向任意方向移动不超过0.2个单位距离都不会移到底面以外的地方去。首先第一步，我们考虑如何求出一个点在一个平面上的投影。我们可以做出该平面的法向量，然后算出该点到平面的距离。然后将该点沿着法向量移动那么长的距离就能求出该点在平面上的投影坐标了。第二步我们想办法检验是否合法。注意到检验的内容又可以等价为投影出来的点到底面的每条边的距离都不小于0.2 个单位距离，于是我们依次求出点到每条边的距离进行判断即可。（如果四点共面就转化为四边形处理）

时空复杂度

时间复杂度$O(1)$

空间复杂度$O(1)$

试题编号

2011 A

题目名称

To Add or to Multiply

题目大意

给出一个在$[p,q]$之间的数，你可以每次把它加上a或者乘以m。现在你需要给出一个最短且字典序最小的操作序列，使得无论给出的是$[p,q]$之间的哪个数，经过操作之后的数都是在$[r,s]$之间的。

算法讨论

首先考虑即使是极限情况能够乘以m的次数也不会很多（$m=1$时特殊处理），所以我们可以枚举究竟乘了多少次。假设我们当前乘了k 次，在不考虑最少次数的情况下，我们通过合法的操作序列能够将p变成的最小的数应该是$m^k+xa>=p$。这个值可以直接通过模的数直接算出来。考虑当前x在m进制下的表示，某一位上是y的话就代表在某次乘法前连加了y次。（这样一定是最小的）考虑我们修改操作序列使得我们的操作序列长度变小并且仍然合法。我们每次修改可以在x对应的m进制数的某一位上加1，但这样会使得答案变大。答案变小的情况只会发生在某一位加上1之后出现了进位的情况。所以我们可以直接枚举将哪一位加进位，并且检验当前操作序列是否合法。这样我们就可以只用做x在m进制下的位数多次的检验即可。

时空复杂度

时间复杂度$O(log^2p)$

空间复杂度$O(logp)$

试题编号

2011 E

题目名称

Coffee Central

题目大意

给出一张方格图，某些位置有咖啡馆。现在每次给出一个询问距离，问从某个位置最多走这么多距离能够到达的咖啡馆的数目最多是多少个。

算法讨论

考虑到每次询问对于一个固定的距离，在原图上从某个点能够走到的图形是一个锯齿形的形状不好处理，所以我们将原图以$(1,1)$为中旋转45度。这样处理之后我们可以发现现在对于一个点询问一个固定距离能够走到的咖啡馆的数目，就是在询问一个矩形内部有多少个咖啡馆。这个可以用前缀和问题解决。至于如何得到旋转后的点的坐标，我们只需要把原坐标(x,y)变化为$(x+y-1,max(n,m)+y-x)$即可。

时空复杂度

时间复杂度$O(QN^2)$

空间复杂度$O(N^2)$

试题编号

2011 H

题目名称

Mining Your Own Business

题目大意

给出一张图，求至少要修几个逃生点才能使得任一个点被删除之后剩下的点仍能够到达某一个逃生点并求方案数。

算法讨论

很容易看出这个问题是跟点双联通分量有关的，所以我们将所有点双联通分量缩点得到了一棵树。由于只会删除一个点，所以这棵树上任何度数不为1的点都是不用修建逃生点的，只需要在度数为1的点中的任意一个非割点上修建一个逃生点即可。所以处理出点双联通分量后扫一遍即可得到答案。

时空复杂度

时间复杂度$O(N)$

空间复杂度$O(N)$

试题编号

2011 I

题目名称

Mummy Madness

题目大意

一个无限大的直角坐标系中你位于$(0,0)$。有N个木乃伊站在其他某些位置。你们都可以在一个单位的时间内移动到周围8个格子中去或者停留在原地。你希望木乃伊在尽量长的时间之后和你相遇，问这个最长的时间。

算法讨论

很显然答案具有单调性，所以首先二分答案。考虑如何检验一个答案m是否可行。这个问题等价于一个$2m*2m$的矩形是否完全被另外N个$2m*2m$的矩形完全覆盖。我们的矩形总共四个方向，我们考虑其中的一个方向。如果N个矩形中的某一个矩形的右下角在第一个矩形之中，那么在第一个矩形之中这个点的左上方一定都被覆盖了（因为所有的矩形的大小是一样的）。那么考虑所有矩形右下角对矩形的覆盖影响就会在该矩形上部形成随着x的增长y也增长的一个轮廓线。于是我们可以使用单调栈维护出这个形状。对于另外的方向也是一样的作法。最后再判断各个轮廓线是否完全覆盖矩形即可。

时空复杂度

时间复杂度$O(N\log N)$

空间复杂度$O(N)$

试题编号

2011 J

题目名称

Pyramids

题目大意

有两种搭建金字塔的方式：将上一层的边长变为比下一层少1或2。现在给出总共的石块数，要求给出一种用上全部石块搭建尽量少且大的金字塔尽量大的方案。

算法讨论

我们发现实际金字塔的大小在整数域内的覆盖范围很广阔，所以对于任意一个正整数都可以用很少的金字塔将其搭建出来，所以可以直接爆搜。（实际上这个答案是不会超过7的）

时空复杂度

时间复杂度$O(N^7)$

空间复杂度$O(N)$

试题编号

2012 B

题目名称

Curvy Little Bottles

题目大意

给出一个函数，求它在某段上的平方值的和以及将其按照某个值分段之后每一段的坐标值。

算法讨论

对于第一问，我们直接将原函数平方之后求出它在该段上的定积分即可。对于第二问，我们可以每次二分下一个分段点的坐标值，然后再用定积分判断即可。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(N)$

试题编号

2012 C

题目名称

Bus Tour

题目大意

有N个城市。你需要定义一个旅行序列，你会按照序列上的序列去遍历完这N个城市，要求从0号城市出发，在$N-1$号城市结束。在从$N-1$号城市返回0号城市时需要重新定义一个旅行序列。假设$H=\lfloor\frac{N-2}{2}\rfloor$，要求在一个旅行序列中前$H$ 个城市编号所形成的的集合与第二个旅行序列后H个城市编号所形成的的集合相等。求所有合法的旅行序列中旅行距离最短的长度。

算法讨论

我们用$f[i][j]$表示从0号城市出发走完集合为i的城市最后达到j城市的最短距离，用$g[i][j]$表示从$N-1$号城市出发走完集合为i的城市最后到达j城市的最短距离。这两个值都可以在$O(2^N*N^2)$的复杂度内解决。然后考虑答案，我们考虑首先我们枚举前H个城市所形成的集合S，然后考虑第一段旅行，由于都是无向边，所以我们可以直接枚举在第一段旅行中前H个城市和剩下的城市进行交界的时候是哪两个城市。同理也可以处理出第二段旅行序列的答案。这样的总复杂度就是$O(2^N*N^2)$。

时空复杂度

时间复杂度$O(2^N*N^2)$

空间复杂度$O(2^N*N^2)$

试题编号

2012 D

题目名称

Fibonacci Words

题目大意

定义01斐波那契串，询问某个串在某个斐波那契串中出现了多少次。

算法讨论

考虑一个斐波那契串对答案的贡献。它是由前两个串拼接而成，那么除了本来就在原来两个串中已存在的答案，新的答案只会在拼接的两部分中出现。由于询问的字符串不会很长，所以我们可以暴力处理出每一个斐波那契串前面以及后面部分的字符串，然后每次拼接时用KMP做出答案即可。

时空复杂度

时间复杂度$O(NL)$

空间复杂度$O(L)$

试题编号

2012 E

题目名称

infiltration

题目大意

N个城市，一个城市被选择之后从它只经过一条边能被到达的城市和它自己都会被标记上。如果从城市i到城市j没有边，那么从城市j到城市i一定有一条边。现在问最少需要选择多少个城市才能使所有城市都被标记上。

算法讨论

由题意可以看出给出的图十分稠密，所以最后的答案一定不会很大，所以直接迭代加深搜索答案即可。（实际上，通过搜索之后发现，在题目给定的数据范围内，最多只用不超过六个点就可以解决问题）

时空复杂度

时间复杂度O(迭代加深层数)

空间复杂度$O(N^2)$

试题编号

2012 K

题目名称

Stacking Plates

题目大意

有N堆圆盘，每次可以将一堆圆盘分成两部分或者合并两堆圆盘，大圆盘必须放在小圆盘下面。问最少的操作次数使得只剩一堆圆盘。

算法讨论

实际上我们每次将圆盘分割出去之后都一定会马上拿回来，并且我们一定是先将所有较小的圆盘处理掉之后才会处理更大的圆盘，所以我们可以按照圆盘大小进行DP。我们用$f[i][j]$表示已经将大小小于等于i的圆盘全部合并并且它们的位置在j堆中的最小代价。那么我们考虑转移的话就是枚举下一个大小的圆盘放在哪一堆之中，每次由上次推算出代价即可。

时空复杂度

时间复杂度$O(N^4)$

空间复杂度$O(N^3)$

试题编号

2012 L

题目名称

Takeover Wars

题目大意

有两个公司有很多子公司，它们每次可以用一个较大权值的子公司消灭对手的一个公司，或者合并自己的两个子公司。问先手是否必胜。

算法讨论

很容易观察出我们最优的策略一定是当前如果自己最大的子公司能够消灭掉对方最大的子公司就消灭否则继续合并。那么只需要将两边的子公司进行排序之后贪心即可。

时空复杂度

时间复杂度$O(NlogN)$

空间复杂度$O(N)$

试题编号

2013 A

题目名称

Self-Assembly

题目大意

有一些正方形，正方形的每条边上都有两个字符，可能是A+、A-、B+、B-、……、Z+、Z-、00这53种中的一种。两个正方形能够被拼接在一起的条件是它们两个用于拼接的两条边是像A+和A-这样能配对字符对。（00和00不能配对）现在给你N种正方形，每种正方形的个数都是无限个，问用这些正方形是否能够实现无限拼接。

算法讨论

如果我们将能够拼接的正方形之间连上边的话，那么问题转换为了一个图里面是否存在环。但是这个算法的复杂度是$O(N^2)$的，不能接受。注意到实际上有用的边的种类只有52种，如果我们将某种边通过拼接后可以转移到的边的种类建立起边的关系。那么，我们问题的本质任然是判断一张图种是否存在环，但是点数已经降至可以接受的范围了。

时空复杂度

时间复杂度$O(N+52^3)$

空间复杂度$O(N)$

试题编号

2013 B

题目名称

Hey, Better Bettor

题目大意

在赌场你每局有p的概率赢1元钱，$1-p$的概率输1元钱。当你的最终受益为负时，赌场会返还你$x\%$的钱，问期望的最大受益。

算法讨论

我们设当我们输到a元或者赢到b元时就不再继续进行赌博，用$f(i)$表示当前受益为i时的最大期望受益。那么我们可以得出$f(a)=x*a$,$f(b)=b$,$f(i)=p*f(i-1)+(1-p)*f(i+1)$。通过这三个式子我们可以解出$f(0)=(((p/(1-p))^b-(p/(1-p))^{b-a})/(1-(p/(1-p))^{b-a}))*b+(1-x)*(1-((p/(1-p))^b-(p/(1-p))^{b-a})/(1-(p/(1-p))^{b-a}))*a$ 那么接下来只需要搜索a和b的值就可以得出我们所需要的答案了。

时空复杂度

时间复杂度$O(N^2)$

空间复杂度$O(1)$

试题编号

2013 C

题目名称

Surely You Congest

题目大意

有一些人需要从某些点走到1号点去，他们都只会沿着最短的路线走。如果在某个时刻两个人需要同时进入同一条路，就会发生矛盾关系，会导致有一个人无法在最短的时间内走到1号点去。问最多能有多少人在最少的时间内走到1号点去。所有人的移动速度相同。

算法讨论

由于所有人都是沿着最短的路线行动，所以先做出最短路图。然后由于到1号点距离不同的人之间一定不会有所影响，所以对每种距离的人做一次网络流算出最大值即可。虽然复杂度很高，但是由于时限很长，所以是能够解决这个问题的。

时空复杂度

时间复杂度$O(NM + N^3M)$

空间复杂度$O(N)$

试题编号

2013 D

题目名称

Factors

题目大意

定义$f(i)$为将i的所有质因子的排列数（比如12有223这3个质因子，有三种排列方式），求$f(i)=x$的最小的i。

算法讨论

有几个显而易见的结论：

1、选择的质数一定是最小的几个质数。

2、将选择的质数从小到大排序后，较小的质数的指数一定大于等于较大的质数的指数。

只有满足上述两个条件，才能够保证做出来的答案最小。利用上述两个条件进行搜索预处理出所有数的答案即可。（答案的取值不会很多）

时空复杂度

时间复杂度O(可行答案数+询问数)

空间复杂度O(可行答案数)

试题编号

2013 E

题目名称

Harvard

题目大意

总共有不超过13个内存，你需要将它们分配给不超过B个部分，每个部分最多拥有S个内存。现在给出一段程序，你需要完成该程序对内存的访问操作。如果访问的是第一部分的内存便可以直接访问，如果不是第一部分的内存，你需要先修改位置到该内存所属于的部分，然后再访问这个内存（如果当前位置就是该部分则不需要修改）。每种操作的代价都为1，程序段的长度为N，求最小的代价。

算法讨论

首先第一步我们需要确定每个内存位置是属于那一部分的，我们用搜索枚举所有可能的情况。注意到除了第一部分剩下的部分都是没有本质区别的，利用这点可以将这部分的搜索大大剪枝，最后该部分的搜索量的极限大概为10^7左右。但是如果在每次搜索后依旧用$O(n)$的复杂度确定答案的话是不能接受的，所以我们需要预处理一些信息。

首先第一步我们枚举哪些内存是属于第一部分的（第一部分的内存一定是越多越好）。我们考虑，对答案的影响在于两个地方，第一个是访问每个变量，第二个是切换部分的代价。第一个代价是定值，所以我们只需要想办法快速算出第二个代价的值。在确定了哪些内存属于了第一部分之后，那么对第二个代价有贡献的只可能是剩下的没有被确定归属的内存。我们可以在每次搜索前提前处理出归属不为第一部分的内存i 的下一个归属不为第一部分的内存是j的个数，记录为$cnt[i][j]$。那么在完成一次搜索的时候，如果i和j 都不属于第一部分并且它们的归属也不同，那么它们就会对答案贡献$cnt[i][j] + cnt[j][i]$。这样我们就把每次搜索时解出答案的复杂度变成了$O(13^2)$，对于$10s$的时限是可以接受的了。

时空复杂度

时间复杂度$O(10^7 * 13^2 + N * 2^{13})$

空间复杂度$O(N)$

试题编号

2013 F

题目名称

Low Power

题目大意

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，每台机器的能量是两个芯片能量的差值。现在给出$2NK$个电池的能量，求所有机器的能量之差的最大值最小的值。

算法讨论

很直观的一点是，如果答案a是合法的，那么对于$bp \geq a$，b也肯定是合法的。所以我们可以二分答案，然后从后往前贪心选取，能选就选，判断是否合法即可。

时空复杂度

时间复杂度$O(2NK * log10^9)$

空间复杂度$O(2NK)$

试题编号

2013 H

题目名称

М а т р ё ш к а

题目大意

有N个玩偶，可以将小的玩偶放到大的玩偶里面去，但是这样需要将大的玩偶拆开，会花费1的代价。两个玩偶合并后变为一个玩偶（但是小的玩偶仍然在大的玩偶内部），每次可以合并两个相邻的玩偶。求最少的代价使得每一个玩偶里面的m个玩偶（包括自己，m不为定值）的大小分别是1、2、3、……、m。

算法讨论

我们另$f[i][j]$表示将i到j这一段的所有玩偶合并为一个玩偶所需要的最小的代价。考虑转移即是枚举一个k，将i到j分为i到k、$k+1$到j两部分，这样的代价即使$f[i][k]+f[k+1][j]$+合并左右玩偶的代价。合并左右玩偶的代价可以重做一次得到。有了f数组，我们可以用$g[i]$表示前i个玩偶合并为最后合法的玩偶的最小代价。转移方式是枚举下一段合并为一个玩偶的位置（需要保证这一段所有玩偶的大小是一个排列）。这样做的复杂度是$O(N^4)$的。由于N最大可以达到500，这样的复杂度不能接受，我们考虑如何优化。复杂度的主要问题来自处理f数组时，由于状态和枚举断点都是难以优化的，所以我们考虑优化算出合并代价的复杂度。考虑我们假设k是i到j中大小最小的位置，那么如果假设k为断点，那么右边的所有的都需要拆开，左边的大于右边的最小值的也都需要被拆开。如果k向左移，我们会发现在移动的过程中代价的变化也是单调的。所以我们用一个单调的指针扫描来维护我们的合并代价，就可以将复杂度降至$O(N^3)$。

时空复杂度

时间复杂度$O(N^3)$

空间复杂度$O(N^2)$

试题编号

2013 J

题目名称

Pollution Solution

题目大意

求一个多边形与一个半圆的面积并。

算法讨论

数据保证了多边形在半圆内部的部分是连续的，所以直接自适应辛普森做即可。

时空复杂度

时间复杂度$O(N*$辛普森迭代次数$)$

空间复杂度$O(N)$

试题编号

2013 K

题目名称

Up a Tree

题目大意

有三段分别用于输出树的前序、中序、后续遍历的递归程序，但是程序内部递归调用的部分可能调用的并非自身而是其他的两种遍历程序中的一种。现在给出按照这三段程序所输出的前序、中序、后序遍历，问所有可能有同样输出的程序段以及正确的前序、中序、遍历。（每个递归程序在所有程序段中被调用的次数一定是两次）

算法讨论

首先第一步是枚举六个调用过程分别调用的是什么，然后我们需要判断当前的调用是否能够输出指定的前序、中序、后序遍历。如果此时存在前序或者后序遍历的话，那么根是可以确定的，我们也可以在中序遍历中找到根的位置然后向左向右递归检验。如果此时都不存在前序和后序遍历的话，那么我们是无法确定根是什么的，所以这个时候我们枚举根是哪一个，这样的话我们就能够确定当前的左子树和右子树再递归检验即可。在代码实现上有比较多的细节，不在这里赘述。注意到由于我们每次需要枚举根的位置，这样有可能会导致同样一个部分的答案被计算了多次，所以我们使用记忆化搜索来优化复杂度。

时空复杂度

时间复杂度$O(26^4 * log26^3)$

空间复杂度$O(26^3)$