第一次作业第二部分
（共 100 题）

广东省中山市中山纪念中学　李成杰

January 13, 2014

Contents

1 ACM/ICPC World Finals 2013

1.1 2013A: Self-Assembly

1.1.1 题目编号

2013 A

1.1.2 题目名称

Self-Assembly

1.1.3 题目大意

有一些种类的正方形，每条边都有标识。

英文字符相同，另外一个字符相异的边可以粘在一起（例如 A+ 能粘上 A-，不能粘上 B-或 A+）。同时有一些边不可能粘上（标识为 00）。

每种正方形有无限个，并可以任意旋转翻转，问是否可能粘出一个无限大的图形。

最多有 40000 种正方形，英文字符只有大写的 26 个，英文字符后的字符只有 + 或 -。

1.1.4 关键词

图论

1.1.5 算法讨论

因为要构建一个无限的图形，而正方形数有限，所以存在循环。

循环的基础是一条循环的链——这条正方形组成的链上，使用到的边的标识组成了循环。

那么由此我们可以构造一个图的模型对应标识的循环。这个图的节点代表 52 个标识。对于每个正方形，每个标识向其他标识的互补标识连一条有向边。这样，图中的环就对应着正方形链上标识的循环。

那么，询问是否可能粘出一个无限大的图形就转变成询问一个 52 个点的有向图上有没有环。在图上遍历一下即可解决。

1.1.6 时空复杂度

- 时间：$O(n)$
- 空间：$O(1)$

1.2 2013B: Hey, Better Bettor

1.2.1 题目编号

2013 B

1.2.2 题目名称

Hey, Better Bettor

1.2.3 题目大意

要求进行若干次赌博。每次赌博输一元或赢一元，赢的概率为 p。假如最后玩家亏本，那么赌场会给回扣，返回输钱数额的 q。求赢钱的最大期望。

$0 \le p < 0.5$、$0 \le q < 1$，p 和 q 均为 10^{-4} 的整数倍。

1.2.4 关键词

数学、二分

1.2.5 算法讨论

既然要赌博若干次，那么考虑结束赌博的条件。

1. 先考虑以时间结束。但是，明显在赌了若干次之后，若输了很多直接走人会让期望变大。所以不可。
2. 赌博中重要的状态只有现在的资产数，那么考虑设置 a 和 b，输了 a 元或赢了 b 元走人。可以发现这是最合理的结束条件。

设 $\lambda = 1 - q$。设 $f(x)$ 为现在的资产为 x 的期望结果。那么有

$$\begin{cases} f(x) = -\lambda a & x = -a \\ f(x) = (1-p) \cdot f(x-1) + p \cdot f(x+1) & -a < x < b \\ f(x) = b & x = b \end{cases}$$

令 $g(0) = f(-a)$，可以见得

$$g(i) = \frac{1}{p} \cdot g(i-1) + \frac{p-1}{p} \cdot g(i-2)$$

如此，g 的特征方程就是

$$-p \cdot x^2 + x + (p-1) = 0$$

解得

$$x_1 = 1, \quad x_2 = \frac{1-p}{p}$$

因为 $g(0) = -\lambda a, \quad g(a+b) = b$，所以令 $\rho = \frac{1-p}{p}$，就有

$$\begin{cases} g(0) = c_0 + c_1 = -\lambda a \\ g(a+b) = c_0\rho^{a+b} + c_1 = b \end{cases} \Rightarrow g(a) = (b + \lambda a) \cdot \frac{\rho^a - 1}{\rho^{a+b} - 1} - \lambda a$$

如此，如果知道 a 和 b，就能够算出期望。

但是怎么确定 a 和 b？打表可以发现，固定 a 或 b，得到的都是单峰函数。那么就可以二分或三分 a 和 b。

二分的上界是一个比较麻烦的问题。因为按题意，$\frac{5001}{4999} \leq \rho \leq 9999$，$\rho$ 大的时候 $a+b$ 不能太大，否则类型就会爆掉；但是 ρ 小的时候 $a+b$ 可能很大。因此要设 $\rho^{a+b} \leq C$，赋 C 一个较大而又不爆类型的值[1]。

1.2.6 时空复杂度

- 时间：$O(\log^2 C)$
- 空间：$O(1)$

[1]据某题解，C 取 10^{10} 即可。

1.3 2013C: Surely You Congest

1.3.1 题目编号

2013 C

1.3.2 题目名称

Surely You Congest

1.3.3 题目大意

有一些人，要从一些城市向 1 号城市进发。这些人只会走最短路，并且在同一时间出发。

要求你保留最多的人，使得存在一个方案让剩下的人不会有至少两个人在同一时间、同一地点向同一条边行进。

路口数 $n \leq 25000$，道路数 $m \leq 50000$，人数 $c \leq 1000$。

1.3.4 关键词

网络流

1.3.5 算法讨论

因为人只会走最短路，所以考虑从 1 出发做一次最短路，求出最短路网络。

在这个最短路网络中，点是分层的。因为在后面几层的点的人不可能追上前面的人，所以堵塞只会在同一层的人之间出现。如此，考虑对每一层做最大流。

明显，源点是人所在的城市，汇点是 1 号点。在同一层中，只要更换源点，不需要还原残量网络。

考虑最暴力的做法：对每一个人做一次 DFS 求增广路。这样，每次 DFS 是 $O(n+m)$ 的，那么复杂度就是 $O(c \cdot m)$，10s 的时限是完全没有问题的[2]。

1.3.6 时空复杂度

- 时间：$O(c \cdot m)$
- 空间：$O(n+m)$

[2]当然，为了快一点可以采用 dinic,SAP 什么的。

1.4 2013D: Factors

1.4.1 题目编号

2013 D

1.4.2 题目名称

Factors

1.4.3 题目大意

设函数 $F(x)$，值为 x 因数分解之后，没有幂运算的乘法表达式的不相同个数。例如 12 有 $2\times2\times3$、$2\times3\times2$、$3\times2\times2$ 三个；因此 $F(12)=3$。

给出若干个 n，求使得 $F(x)=n$ 的最小 x。数据组数 ≤ 1000，$n<2^{63}$，保证解 $x<2^{63}$。

1.4.4 关键词

搜索

1.4.5 算法讨论

- 因为要使 x 最小，所以因数分解之后，素数越大，指数不增。
- 因为 $21!>2^{63}$，所以素数个数不超过 20，大部分情况更少。
- 由于上述限制比较严，所以可以猜想满足限制的状态并不多。
- 事实上，写程序验证之后，可以发现状态数不超过 4 万。
- 于是，就可以读入所有询问，然后搜索一遍得出答案。

1.4.6 时空复杂度

- 时间：$O(?)$
- 空间：$O($询问数$)$

1.5 2013E: Harvard

1.5.1 题目编号

2013 E

1.5.2 题目名称

Harvard

1.5.3 题目大意

这道题每个数据会给出一个程序，这个程序只有循环语句和变量访问语句。

这些程序只有变量访问耗费时间。变量都是静态的，存储在 B 个容量为 S 的存储器中，存储器编号从 0 开始。

每次变量访问，可以直接访问 0 号存储器或第 BSR 号存储器；BSR 值的改变需要一个时间单位，一开始 BSR 值并不存在。访问存储器中的变量也需要一时间单位。

你可以随意安排变量的存储位置，求程序运行的最短时间。

$1 \leq B, S \leq 13$.1 $\leq$ 变量数 $\leq \min(B \cdot S, 13)$，单个循环的次数不超过 10^6，循环体非空，程序访问变量的次数不超过 10^{12} 次。程序长度 ≤ 1000.

1.5.4 关键词

搜索、预处理

1.5.5 算法讨论

- 首先，无论如何安排存储位置，变量的访问次数是不变的，变化的只有 BSR 的赋值次数。那么首先预处理出变量访问次数。这一步可以用 $O(Len^2)$ 的递归做，也可以用 $O(Len)$ 的模拟栈做。
- 因为 BSR 要尽量少改变，所以就要将 0 号存储器放满。因此，我们要搜索哪些变量放到 0 号存储器。因为要放满，所以这一步的复杂度是一个组合数。
- 搜索出一个方案之后，我们就可以无视掉程序中访问存放在 0 号存储器的变量的操作，处理出一个二维数组 $C(i, j)$，表示访问 i 变量然后紧接着访问 j 变量的次数。这一步的预处理需要用模拟栈做到 $O(Len)$。
- 因为两个变量可能在两个不同的循环里，所以在最开始要预处理任意两个位置之间最少的循环嵌套层数。

- 处理出 $C(i,j)$ 之后，搜索剩下的变量放在哪些存储器。这步搜索除了最优性剪枝，还要注意把没有变量的多个存储器视为一个。

1.5.6 时空复杂度

令变量有 V 个，给出程序长度为 Len.

- 时间：$O(C(V,S)\cdot B^{V-S}+C(V,S)\cdot Len)$
- 空间：$O(Len)$

1.6 2013F: Low Power

1.6.1 题目编号

2013 F

1.6.2 题目名称

Low Power

1.6.3 题目大意

有 n*k*2 个数，要分成 n 组，每组再分为两部分（大小均为 k）。一组的差异值为这组中两部分各自最小值的差。求所有组最大差异值的最小值。

$2nk \leq 10^6, 1 \leq$ 任一个数 $\leq 10^9$.

1.6.4 关键词

二分、贪心

1.6.5 算法讨论

因为差异值为两部分各自最小值的差，所以假如将各组的两个最小值挑出来排序，同一组的两个值是相邻的。证明显然：假如不相邻，交换所属无论如何不会变差。

因此我们可以按照最小值的大小对这些组排序。那么让所有数从小到大排好序，第一组的最小值必是第 1、2 个数，第二组的最小值必在第 3 到 $2k+2$ 个数中，第三组的最小值必在第 5 到第 $4k+2$ 个数中…

如此我们可以想到一个结论：若答案不大于 L，那么令 $f(x)$ 为 1 到 x 个数中不大于 L 的数的个数，即有充要条件 $f[2 \cdot (i-1) \cdot k + 2] \geq i, i \in [1, n]$. 必要性如上，充分性：用贪心即可构造出一个解。

这样，我们就有一个算法：先排序，然后二分答案，利用上述结论判断。

1.6.6 时空复杂度

- 时间：$O(n \log_2 n)$
- 空间：$O(n)$

1.7 2013H: Матрёшка

1.7.1 题目编号

2013 H

1.7.2 题目名称

Матрёшка

1.7.3 题目大意

你需要对一些俄罗斯套娃进行合并，使得最终任一套娃（集）的套娃大小连续且有大小为 1 的套娃。

一开始套娃都是单个的，并且排成一行。你合并的时候只能合并相邻的套娃(集)，并且合并好的套娃集不能拆开再分别与其他套娃合并。

合并套娃的时候只有打开套娃耗费时间，打开任一套娃只耗费 1 时间单位。

问最短的时间。

套娃大小最小为 1. 套娃数量 ≤ 500.

1.7.4 关键词

动态规划

1.7.5 算法讨论

我们可以用平方时间算出哪些区间可以合并出最终的套娃（集）。

观察可以发现，$[a,b]$ 与 $[c,d]$ 所表示的套娃集合并的时间等于在另一个区间有更小的数的数的个数，即大于另一个区间最小值的数的个数。

那么我们令$g(i,j)$ 表示把区间 $[i,j]$ 内的套娃合并成一个套娃集所需的时间。如此我们需要枚举一个变量 k，合并区间 $[i,k]$ 和 $[k+1,j]$ 来得到 $g(i,j)$ 的最小值。

令第 p 个数为区间 $[i,j]$ 的最小值，那么让 k 从 $p-1$ 枚举到 i 或从 p 枚举到 j；最小值不在的区间中所要打开的套娃数等于区间大小，而另一个区间耗费的时间在得知数的大小顺序之后能均摊 $O(n)$ 维护——最小值所在的区间要维护大于另一区间最小值的数的个数，另一区间要维护最小值。在区间 $[i,j]$ 内的数排序之后，记录下每个数的位置，就能用指针维护上述两个值了。

得到 g 之后，我们用$f(i)$ 表示 1 到 i 最终合并成若干个合乎结束条件的套娃集的时间，那么就有 $f(i)=\min\{f(j-1)+g(j,i)\}, j\in[1,i], \exists k\in Z_+$ 使得

$\{x|x = A_p, j \leq p \leq i\} = \{x|1 \leq x \leq k, x \in Z\}$. 最后 $f(n)$ 就是答案。

1.7.6 时空复杂度

- 时间：$O(n^3)$
- 空间：$O(n^2)$

1.8 2013I: Pirate Chest

1.8.1 题目编号

2013 I

1.8.2 题目名称

Pirate Chest

1.8.3 题目大意

Dick 要在一个 $n \times m$ 的池子里放入一个底面最大为 $a \times b$ 的箱子，箱子高度任意。为了隐藏箱子，箱子顶部要严格低于水面。池子四周是陡峭笔直的悬崖，放入箱子后水面会自然上涨。给出每个格子的深度，和四个参数，求箱子最大的体积。

$1 \leq a, b, n, m \leq 500$，$0 \leq d_{i,j} \leq 10^9$。

1.8.4 关键词

单调数据结构、枚举

1.8.5 算法讨论

如果，现在放入的箱子底面为 $w \times l$，覆盖的水域最小深度为 d，那么高度 $h < \frac{nmd+wlh}{nm}$。移项可得 $h < \frac{nmd}{nm-wl}$。

为了让体积最大化，明显 $h = \lfloor \frac{nmd-1}{nm-wl} \rfloor$。

考虑枚举 w 和第一行，那么就要在深度不变下使 l 最大——因为 l 增大，h 也会增大。

如此，选定一列的深度为最小深度之后，就可以向两边扩展，得到最大的 l。计算两边扩展的长度时可以 m 列一起计算，用单调栈维护，找到左右最近的更小深度的位置。然后就可以算出 h 和体积。

那么，剩下的问题就是计算每一列 w 行的最小深度。设 $C(k, i, j)$ 表示 $w = k$，第 j 列，i到$i + k - 1$ 行的最小值，那么就可以 $O(1)$ 的转移。实际上存储的时候可以省掉 k。

1.8.6 时空复杂度

- 时间：$O(n^3)$[3]
- 空间：$O(n^2)$

[3]译者非常丧心病狂的卡常数——时限 3s，他用了 2.2s

1.9 2013J: Pollution Solution

1.9.1 题目编号

2013 J

1.9.2 题目名称

Pollution Solution

1.9.3 题目大意

有一个在 X 轴及其上方的简单多边形和一个以圆心为原点半径为 r 的圆，求圆与多边形重合面积。

多边形边数 $n \leq 100$。

1.9.4 关键词

计算几何

1.9.5 算法讨论

首先求出多边形与圆弧的交点 (用二分、解方程什么的)，这样重合部分就可以用圆弧、直线来描述。

把求出的交点加入多边形顶点点集，然后按逆时针扫描点集。对于两个相邻的点，若其中一个点在圆外，那么这两个点就代表了一段圆弧，并且这两个点分别在扇形的两条半径的延长线上；假如没有，那么这两个点代表了一条线段。如图1：

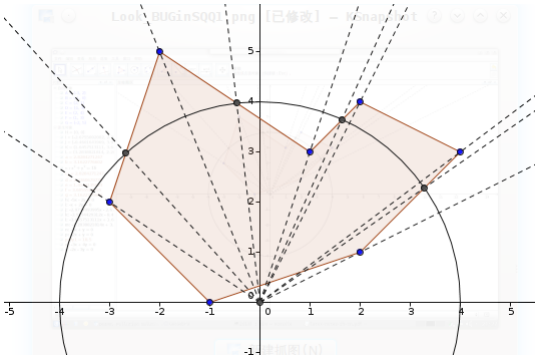

Figure 1: 示例

然后我们就可以模仿叉积求多边形的面积的方法：对于一条线段，加上原点到两个端点的叉积；对于一段圆弧，加上扇形的有向面积。以上都是逆时针方向为正，顺时针方向为负。

最后加起来的值就是重合部分的面积。证明和叉积求多边形面积类似。

1.9.6 时空复杂度

因为一条线段跟圆弧最多有两个交点，所以最后处理的点数是 $O(n)$ 级别的。

- 时间：$O(n \log C)$
- 空间：$O(n)$

1.10 2013K: Up a Tree

1.10.1 题目编号

2013 K

1.10.2 题目名称

Up a Tree

1.10.3 题目大意

有一位新生写了三个函数，为树的前序、中序、后序遍历。但是其中调用函数的语句写错了。一个可能的错误情况如下：

```
void prePrint(TNode t)
{
    output(t.value);
    if (t.left != null)
    prePrint(t.left);
    if (t.right != null)
    postPrint(t.right);
}

void inPrint(TNode t)
{
    if (t.left != null)
    inPrint(t.left);
    output(t.value);
    if (t.right != null)
    postPrint(t.right);
}

void postPrint(TNode t)
{
    if (t.left != null)
    inPrint(t.left);
    if (t.right != null)
    prePrint(t.right);
    output(t.value);
}
```

同时，他犯的错误满足一下规律：

1. 除了调用函数的部分其他没错。
2. 调用的语句总是有两个前序，两个中序，两个后序；错误的只是位置。

现对这个错误程序进行了测试，输入了一棵字符树。字符树每个节点均为一个不重复的大写英文字母。只给出错误程序的输出，求可能的错误情况、和每种错误情况下前中序遍历字典序最小的输入字符树。

输入字符树的节点数 ≤ 26。

1.10.4 关键词

记忆化搜索

1.10.5 算法讨论

因为调用入口只有六个，每种调用各两个，所以所有调用情况有 $C_4^2 \cdot C_6^2 = 90$ 个。于是，就可以暴力枚举所有调用情况然后再计算。

确定了调用的情况之后，就要搜索可能的字符树。第一步中，根据错误前序、中序，可以得知根和左右子树大小。接着可以递归到左右子树搜索，再合并。但是，因为调用情况的关系，递归中有可能只剩下前中后序中的一两个。如此，就要分类考虑一下情况：

- 有中序，前后序至少有一个：*通过前后序中的一个，可以得知根；再通过中序可以知道左右子树的大小；这种情况转移就是 $O(1)$ 的了。*
- 只剩下了中序：*枚举根的位置，就可以得出左右子树大小，并递归计算；这样，转移为 $O(n)$。*
- 没有中序：*因为通过前后序之一可以得知根，所以枚举左子树大小，就可以递归计算；转移同为 $O(n)$。*

确定了搜索方法之后，考虑时间问题。加上记忆化之后，Key 是三个区间，所以 Key 的数量为 $O(n^6)$ 级别。再加上每次搜索至多 $O(n)$ 的转移，和枚举调用情况的复杂度之后，就有时间上界 $O(90 \cdot n^7) \gg 10^9$。

这个复杂度看上去很可怕，但是在加上判断合法性的剪枝之后，实际的复杂度远小于这个上界；就可以过了。

1.10.6 时空复杂度

- 时间：$\ll O(90 \cdot n^7)$
- 空间：$\ll O(n^6)$

2 ACM/ICPC World Finals 2012

2.1 2012A: Asteroid Rangers

2.1.1 题目编号

2012 A

2.1.2 题目名称

Asteroid Rangers

2.1.3 题目大意

在三维空间上有很多个点，以恒定的速度移动。它们之间连边的费用就是欧几里德距离。求从零时刻开始，连边情况不同的最小生成树有多少个。

点数 ≤ 50，初始位置坐标绝对值 ≤ 150，速度矢量坐标绝对值 ≤ 100。

2.1.4 关键词

暴力、最小生成树

2.1.5 算法讨论

明显，各种最小生成树的过渡时刻里，有若干条边相等。那么考虑解一元二次方程计算出所有任意两条边相等的时刻；排序然后求 $t + 0.5 \times 10^{-6}$ 的最小生成树并判断是否改变了。

但是任意两条边相等的时刻的数量是 $O(n^4)$ 级别的，上面的算法时间复杂度为 $O(n^6)$，铁定超时。

考虑增加剪枝；对于某些边相等的时刻，若最小生成树改变，那么存在两条边在这个时刻相等，且后来变劣的那条边在上一个唯一的最小生成树上、变优的那条边不在。

因为边是 n^2 的，最小生成树只有 $n - 1$ 条边，所以实际效果约相当于指数减一。虽然理论的最劣时间还是 $O(n^6)$，但是实际效果相当于 $O(n^5)$，就可以过了。

2.1.6 时空复杂度

- 时间：$\leq O(n^6) \quad \approx O(n^5)$

- 空间：$O(n^4)$

2.2 2012B: Curvy Little Bottles

2.2.1 题目编号

2012 B

2.2.2 题目名称

Curvy Little Bottles

2.2.3 题目大意

Jill 需要在一个奇怪的瓶子上画刻度。因此她需要知道装入液体体积为 $k \cdot \Delta$ 时，液面距离瓶底的高度。

瓶子是通过一个由 x 轴的一部分、垂直 x 轴的两条线段、一个多项式函数的一段组成的图形，绕 x 轴旋转而成。瓶底和瓶口均为垂直 x 轴的线段旋转而成，且瓶底为靠左的那条线段。

给出多项式 P，两条线段的 x 坐标，求瓶子的体积和前八个刻度离瓶底的距离。

多项式次数 $n \leq 10$，每项系数绝对值 ≤ 100，横坐标绝对值 ≤ 100，两线段距离 > 0.1，$1 \leq \Delta \leq 500$，所有标记与瓶子顶部的距离不会小于 0.01，瓶子的体积不超过 1000，四舍五入后任意两个标记间的距离至少为 0.05，对于任意 $x \in [x_l, x_r], P(x) > 0$。

2.2.4 关键词

数学

2.2.5 算法讨论

先考虑计算瓶子的体积。

- 因为一条条线积分就算出了面积，所以考虑类比推广到面与体的情况。
- 考虑垂直 x 轴将瓶子切成无数个面，那么就可以通过对这些面积分求体积。
- 因为每一个切出来的面，都是由多项式函数上的一点旋转而成，所以面为半径等于点高的一个圆。
- 如此，就得到

$$V = \int_{x_l}^{x_r} \pi P(x)^2 \cdot \mathrm{d}x$$

那么瓶子的体积求出来了；同时，可以发现，通过改变积分的区间，能求出任意高度液面的体积。

于是，就可以采用二分的方式，二分得出每个刻度体积的高度。

2.2.6 时空复杂度

- 时间：$O(n^2 + n \log C)$
- 空间：$O(n)$

2.3 2012C: Bus Tour

2.3.1 题目编号

2012 C

2.3.2 题目名称

Bus Tour

2.3.3 题目大意

给出 h 个旅馆，1 个总部，1 个景点。要求从总部经所有旅馆再到景点，再经所有旅馆回到总部。经过旅馆可以让客人上下车；但是去程里前 $\lfloor \frac{h}{2} \rfloor$ 个下车的旅馆，要在回程里也成为前 $\lfloor \frac{h}{2} \rfloor$ 个下车的旅馆。求在此限制下回到总部的最短时间。$h \leq 18$，任一条路需要时间 ≤ 3600.

2.3.4 关键词

状压 DP

2.3.5 算法讨论

因为点数很少，所以一个很自然的思路就是状态压缩 DP。

用 $f(s,i)$ 表示从总部出发，旅馆的经过状态为 s，最后一个点为 i 的最短时间；用 $g(s,i)$ 表示从景点出发，s 与 i 的意义和 $f(s,i)$ 一样。如此，假如去程前 $\lfloor \frac{h}{2} \rfloor$ 经过的旅馆集合为 S，另旅馆全集为 C，那么去程的时间为 $f(S,i)+dis(i,j)+g(C-S,j)$，回程的时间为 $g(S,i')+dis(i',j')+f(C-S,j')$.

这样，我们只要用 Floyd 预处理 dis，然后 DP 出 f 和 g，就可以枚举 S,i 和 j 算出总时间。

2.3.6 时空复杂度

- 时间：$O(h^2 \cdot 2^h)$
- 空间：$O(h \cdot 2^h)$

2.4 2012D: Fibonacci Words

2.4.1 题目编号

2012 D

2.4.2 题目名称

Fibonacci Words

2.4.3 题目大意

定义 $+$ 为字符串的连接，那么斐波那契 01 字符串的定义如下：

$$F(n) = \begin{cases} 0 & n = 0 \\ 1 & n = 1 \\ F(n-1) + F(n-2) & n > 1 \end{cases}$$

给定一个模式串 p，和一个数 n，问 p 在 $F(n)$ 中出现了多少次。

$n \leq 100$，$|p| \leq 10^5$。

2.4.4 关键词

字符串、递归

2.4.5 算法讨论

明显，从 $F(n-1)$ 变换到 $F(n)$，有下面的规律：

$$\begin{cases} 0 \to 1 \\ 1 \to 10 \end{cases}$$

Proof. 由定义，当 $n \geq 3$ 时，$F(n-1) \to F(n)$ 可以拆分为 $F(n-2) \to F(n-1)$ 和 $F(n-3) \to F(n-2)$。那么递归地拆分下去，就会得到若干个 $F(0) \to F(1)$ 和 $F(1) \to F(2)$，也就是上述规律。 □

利用这一规律，考虑将 p, n 递归回更小的 $p', n-1$ 统计答案。当 $|p'| = 1$ 时，答案就是一个斐波那契数。

但是，对于 p 末尾的 1，我们并不知道它是怎么得到的，因此递归就有分支。

另外，由变换的规律，可以发现，合法情况下不能有相邻的 0、或连续 3 个的 1. 这样，不合法分支就可以剪去。

若每次递归都有分支，下一层递归的规模是 $fib(k-1)$，那么估算为 $T(n) = 2T(0.618 \cdot n) + O(n)$，由主定理得到估算复杂度约为 $O(n^{1.4})$。因为事实上剪枝的效果很好，所以实际的复杂度远小于这个值。

2.4.6 时空复杂度

- 时间：$\ll O(|p|^{1.4})$
- 空间：$O(n \cdot |p|)$

2.5 2012E: infiltration

2.5.1 题目编号

2012 E

2.5.2 题目名称

infiltration

2.5.3 题目大意

给出一个有向图。任意两个点之间有一条有向边。问：取出最少的点，使得图中任意一点都，是取出的点或存在一个取出的点能一步到达它。点数 ≤ 75.

2.5.4 关键词

搜索

2.5.5 算法讨论

可以用一个很简单的证明证明答案不会超过 6 个点。思路如下：一个 n 个点的图，一共有 $C(n,2)$ 条边，最坏情况下，出度最大的一个点的出度为 $\lceil \frac{C(n,2)}{n} \rceil = \lceil \frac{n-1}{n} \rceil = \lfloor \frac{n}{2} \rfloor$. 那么我们贪心地取出度最大的点，剩下的点形成一个子问题，规模变成 $n - \lfloor \frac{n}{2} \rfloor - 1$.

这样，$n = 75$ 的情况下采用贪心策略，答案最多不超过 6 个点。因为这是贪心策略，所以实际上答案上界会更紧一些。考虑到搜索的效果，采取迭代搜索加位压优化是比较明智的策略。

事实上，存在一个结论声称答案不超过 5，但是证明据说十分繁琐难懂[4]。因此搜索的叶子约有 $C(75,5) = 17259390$ 这种数量，位压优化后可以过。

2.5.6 时空复杂度

- 时间：$O(C(n,5))$
- 空间：$O(n^2)$

[4]如果不知道结论，若搜索 5 步没有答案，就可以直接调用贪心。

2.6 2012I: A Safe Bet

2.6.1 题目编号

2012 I

2.6.2 题目名称

A Safe Bet

2.6.3 题目大意

有一个 $R \times C$ 的网格，某些格子上有与格子线夹角成 45° 的镜子。镜子两面都可以反射。

一束激光从格子 $(1,1)$ 左边向右射入网格。问激光能否在格子 (R,C) 向右射出网格，若不可，那么只增加一个镜子时，有多少个位置可以实现目标，并在有至少一个位置时输出字典序最小的位置。

$1 \le R,C \le 10^6$，设格子 $(1,1)$ 在左上角，那么有 m 面/形状的镜子，n 面 \ 形状的镜子。$1 \le n,m \le 2 \times 10^5$。

2.6.4 关键词

数据结构

2.6.5 算法讨论

首先，判断不需要增加镜子的情况：将所有镜子分别以 x 坐标优先和 y 坐标优先排两次序；那么，对于激光的每一次射出或反射，就可以二分找出下一个照射的镜子。

需要镜子时，考虑从 (R,C) 出发，反向射出一束激光。那么前后两束激光路径的相交处就是可行的位置。

将激光的路径分开成一条条线段。那么就是求两类与坐标轴垂直的线段之间的交点数。这是一个经典问题。比较好想的是扫描法。

- 按 x 坐标扫描，用一个树状数组维护每个 y 坐标下与扫描线垂直相交的另类线段数。
- 每扫到一个与扫描线重合的己类线段，就可以把区间和加入答案数，然后在树状数组上二分找到 y 坐标最小的相交位置，并更新答案位置。
- 最后交换类别再做一次即可。

2.6.6 时空复杂度

- 时间：$O((n+m)\log C)$
- 空间：$O(n+m+C)$

2.7 2012K: Stacking Plates

2.7.1 题目编号

2012 K

2.7.2 题目名称

Stacking Plates

2.7.3 题目大意

现在有一堆堆盘子，需要合并成一堆。

有两种操作：

- 拆分：可以将一个盘堆堆顶任意数目的盘子抬起，然后放置在地上，形成新的一堆盘子，使堆一分为二。
- 合并：可以将一个盘堆放置在另一堆的堆顶，前提是在上方的堆最底层的盘子尺寸不大于在下方的堆最顶层的盘子尺寸。

求使用这两种操作的最小总次数。

初始时，堆数 n 和每堆的盘数 ≤ 50；数据组数 $T \leq 400$。

2.7.4 关键词

动态规划

2.7.5 算法讨论

显然，最优时，存在一个方案必定是执行完所有拆分操作之后再合并。

Proof. 把每堆中相邻两数的相邻关系称为“键”。那么操作就是断键和生成键。

最优方案中，不会有在同一个地方同时有断键和生成键，因为通过同时减去两种操作可使得方案更优。因此，被断的键的只会是初始的键，也就是说可以将所有断键操作移到最前，并不改变方案合法性。 □

那么，如果没有相同大小的盘子的话，目标状态是固定的。这样就可以贪心的判断一个键是否需要断掉。

但是有相同的盘子时，相同大小盘子间的顺序在目标状态是不固定的。通过观察，可以发现，最优情况时：

1. 初始时，在同一堆的盘子间的键不会断开。
2. 在这些盘子中，有意义的信息只是两端的盘子属于初始时刻的哪一堆。

那么，就可以设 $f(i,j)$ 表示，DP 到第 i 小个数，最后一个数属于初始时刻的第 j 堆时，保留的最大键数；同理，$g(i,j)$ 表示第一个数属于初始时刻的第 j 堆。如此，转移就是

$$g(i,j) = \max(\{f(i-1,k)|k \neq j\} + \{f(i-1,j)+1\}) + V_i$$

$$f(i,j) = \begin{cases} \max\{g(i,k)|k \neq j\} & |j| > 1 \\ g(i,j) & |j| = 1 \end{cases}$$

因为转移的时候，只有 j 和 k 是否相同的两种情况，所以转移中有用的只有最大值和次大值[5]。那么就可以处理出最大值和次大值，然后每次判断并转移的复杂度就是 $O(1)$ 的了。每次 DP 的复杂度降为 $O(n^3)$，就可以过了。

2.7.6 时空复杂度

- 时间：$O(T \cdot n^3)$
- 空间：$O(n^3)$

[5]注意，有可能相同。

2.8 2012L: Takeover Wars

2.8.1 题目编号

2012 L

2.8.2 题目名称

Takeover Wars

2.8.3 题目大意

有两个公司，他们有很多子公司。因为战争，所以他们轮流采取两种操作：

- 合并：合并己方的两个子公司，新公司的实力为原两公司之和。
- 攻击：选取己方的一个子公司，让敌方的一个实力小于选取公司的子公司消失。

问先手胜还是后手胜。

每方子公司数 $\leq 10^5$，每个子公司的初始实力 $\leq 10^{12}$。

2.8.4 关键词

博弈、贪心

2.8.5 算法讨论

明显，有以下两个贪心的规律：

1. 攻击时，必需要攻击对方最大的子公司。
2. 合并时选存活的最大的两个合并。

由上述规律，更进一步地，有：若一方进行过合并操作，被攻击后会输。

Proof. 被攻击之后，选取剩下两个最大的合并的结果并不会大于之前合并的结果。所以无论怎么合并都会被对方消灭。 □

再进一步思考，可以发现真正有选择的只有先手的第一步。

Proof. 若第一步合并，那么就可以利用上述推论推导胜负。若第一步攻击，后手下一步只能合并，与上一情况同理。 □

由此，算法就是对子公司的实力排序，枚举先手的第一步，然后利用推论 $O(n)$ 判断胜负。

2.8.6 时空复杂度

- 时间：$O(n \log n)$[6]
- 空间：$O(n)$

[6]若用基排可以 $O(n)$.

3 ACM/ICPC World Finals 2011

3.1 2011A: To Add or to Multiply

3.1.1 题目编号

2011 A

3.1.2 题目名称

To Add or to Multiply

3.1.3 题目大意

题目给出两种操作，对数加上 a 或乘上 m。求最短的一系列操作使得任意在区间 $[p, q]$ 内的数经过这一系列操作都落在 $[r, s]$ 区间内。

3.1.4 关键词

贪心

3.1.5 算法讨论

明显，操作结果可以用式子 $\{\ldots[(x+a+a+\ldots)\cdot m+a+a+\ldots]\ldots+a\}\cdot m+a+a+\ldots$ 来表示。于是我们把式子拆开，就变成了

$$(x+c_k a)m^k+\sum_{i=0}^{k-1}c_i a m^i$$

这样，我们就可以枚举 k，然后贪心算出 c 数组。枚举了 k 之后，从高位到低位做，算出不用后面的位时 c_i 的上下限；若可行，则取最小值结束，否则取最大的可行值继续。这是因为高位如果小了，低位数值的增加相比很大。算 c 数组的时候同时也要判断这个 k 值是否可行。还有就是当 $m=1$ 的时候要特判一下。

3.1.6 时空复杂度

- 时间：$O(\lg^2 n)$
- 空间：$O(\lg n)$

3.2 2011C: Ancient Messages

3.2.1 题目编号

2011 C

3.2.2 题目名称

Ancient Messages

3.2.3 题目大意

给出一个扫描得出的 01 矩阵，0 表示白色，1 表示黑色。要求辨识一些象形文字各个的个数。

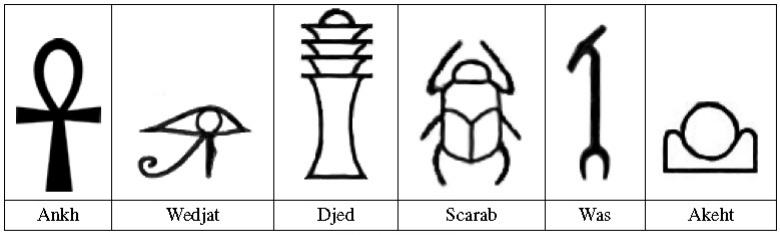

图1：六个象形文字

Figure 2: 题目提供的图片

01 矩阵遵循：

1. 图像仅包含上图所示的象形文字。
2. 每个图像至少有一个有效的象形文字。
3. 每个黑色像素都是一个有效象形文字的一部分。
4. 每个象形文字由一个联通的黑色像素块组成，对于每个黑色像素，至少在其上下左右至少有一块黑色像素块。
5. 象形文字互不接触，而且不存在一个象形文字在另一个的内部。
6. 如果有两块黑色像素对角线相接处，则必然存在一块公共接触的黑色像素。
7. 象形文字可能会扭曲，但是它的拓扑结构必然会和上图所示的一个等价。(如果一个图像可以通过拉伸但不被毁坏的方式转化成另一个，则它们是拓扑等价的)

矩阵最多 $n \leq 200$ 行，每行 $w \leq 50$ 个字符。

3.2.4 关键词

染色

3.2.5 算法讨论

观察图2，可以发现，每个象形文字内部的白色连通块数目都不一样。

那么，就有一个基于 DFS 的染色方案：

- 用 DFS 把所有白色连通块标上序号。
- 找到每个黑色连通块，DFS 染色时顺便检查相邻的白色连通块有多少个。
- 根据相邻白色连通块数量，可以判断黑色连通块是什么文字，然后增加相应的统计量。

3.2.6 时空复杂度

- 时间：$O(nw)$
- 空间：$O(nw)$

3.3 2011E: Coffee Central

3.3.1 题目编号

2011 E

3.3.2 题目名称

Coffee Central

3.3.3 题目大意

一个网格，一些格点是特殊的。若干次询问，问哪个点曼哈顿距离不大于 m 内有最多的特殊点。网格大小 $\leq 1000 \times 1000$，特殊点数 $\leq 5 \times 10^5$，询问数 $Q \leq 20$。

3.3.4 关键词

暴力

3.3.5 算法讨论

一个明显的思路就是，枚举中心点，计算曼哈顿圆内特殊点的个数。

但是直接统计曼哈顿圆很恶心。因此考虑将曼哈顿距离转化为切比雪夫距离。将原坐标 (x, y) 变成 $(x + y, x - y)$ 即可。

这样，预处理一个二维数组存储前缀和，就能每次 $O(1)$ 算出切比雪夫圆内特殊点个数。每次询问暴力枚举答案即可。

3.3.6 时空复杂度

- 时间：$O(Q \cdot L^2)$
- 空间：$O(L^2)$

3.4 2011F: MachineWorks

3.4.1 题目编号

2011 F

3.4.2 题目名称

MachineWorks

3.4.3 题目大意

有一些物品，只能在第 D_i 天购买，需要金钱 P_i，卖出的价格为 R_i，若放着它一整天不进行买卖操作则收益 G_i。同时只能持有一件物品。在 $T+1$ 天，会强制卖掉持有的物品并结算收益。中途资产不能为负。

物品数 $n \leq 10^5$，$T \leq 10^9$，$1 \leq D_i \leq T$，$1 \leq R_i < P_i \leq 10^9$，$1 \leq G_i \leq 10^9$。

3.4.4 关键词

斜率优化、DP

3.4.5 算法讨论

很容易可以看出一个 $O(n^2)$ 的 DP。设 $f(i)$ 表示刚买第 i 个物品时的最大收益，那么有 $f(i) = \min\{f(j) + G_j \cdot (D_i - D_j - 1)\}$。

考虑优化转移。观察方程，假如在一次决策中选 i 比 j 更优，且 $G_i > G_j$，那么

$$(k - D_i) \cdot G_i + f(i) > (k - D_j) \cdot G_j + f(j)$$
$$-k < \frac{[f(i) - D_i \cdot G_i] - [f(j) - D_j \cdot G_j]}{G_i - G_j}$$

设 $x_i = G_i$，$y_i = f(i) - D_i \cdot G_i$，那么就可以发现最优决策是凸壳上的一个点。

但是 G_i 和 D_i 并没有什么联系，所以做法比较简单的有两种。

- 用数据结构维护凸壳。因为切线的斜率单调递减，所以决策点的移动是 $O(n)$ 级别的。于是就可以暴力维护决策点。或者二分找到决策点。
- 按 D 分治，DP 完前一半之后结算前一半对后一半的贡献。这时就可以直接用归并排序和栈线性构造出凸壳，然后用上面暴力维护决策点的方法就可以做到 $O(n \log n)$ 的复杂度。

3.4.6 时空复杂度

- 时间：$O(n \log n)$
- 空间：$O(n)$

3.5 2011H: Mining Your Own Business

3.5.1 题目编号

2011 H

3.5.2 题目名称

Mining Your Own Business

3.5.3 题目大意

给出一个图，要求在一些点上放标记，使任意一个点删掉后剩下的每个连通块至少有一个点有标记。求最少标记数和最少标记情况下放标记的方案数。

点数 $n \leq 50,000$，边数 $m \leq 50,000$。

3.5.4 关键词

割点、图论

3.5.5 算法讨论

首先，我们很明显不会把标记放在割点上；同时一个点双连通分量内只需要至多一个标记。

将割点从各个点双连通分量删去后，任意两个点双连通分量之间没有重合点；于是我们在观念上把每个删去割点的点双连通分量抽象成一个个点，这样原图就成为了一棵无根树。

观察可以发现，无根树只有叶子需要放标记。那么我们要在每个在抽象树中代表叶子的点双连通分量中放一个标记。因为点双连通分量通过割点与其它分量连通，所以“叶子”分量中至多有一个割点。这样，最少标记数就能轻易统计出来，方案数就是所有“叶子”分量除去割点的点数的乘积。同时这样也有一个好处，就是只用求割点，求方案数的时候遍历一下分量就好。

特别的是，若原图点双连通并且点数大于一，要取两个标记。

3.5.6 时空复杂度

求割点用 trajan，就能线性时间了。

- 时间：$O(n+m)$

- 空间：$O(n+m)$

3.6 2011I: Mummy Madness

3.6.1 题目编号

2011 I

3.6.2 题目名称

Mummy Madness

3.6.3 题目大意

有很多个木乃伊在追你。

这幅广阔的地图有很多方格，木乃伊和你在其中一些方格内。经过一秒的时间，你和木乃伊能移动到相邻的 8 个方格内。木乃伊总是往最接近你的方向移动。问木乃伊最迟的把你追到手的时间。

坐标绝对值 $\leq 10^6$，$0 \leq$ 木乃伊数 $\leq 10^5$.

3.6.4 关键词

二分、扫描线、线段树

3.6.5 算法讨论

二分答案 k 之后，任意一人的行动范围都是边长为 $2 \cdot k + 1$ 的正方形。然后就要求所有木乃伊的正方形是否把你的正方形覆盖。因此采用扫描线算法把平面问题转化为一维的问题。

扫描时，用线段树维护扫描线上，你的正方形内，每个位置被多少个木乃伊正方形覆盖了；同时维护这个值的最小值，就能知道这条扫描线上有没有没被木乃伊正方形覆盖的地方。

注意常数。

3.6.6 时空复杂度

令 L 为地图大小。

- 时间：$O(n \log^2 L)$
- 空间：$O(L)$

3.7 2011J: Pyramids

3.7.1 题目编号

2011 J

3.7.2 题目名称

Pyramids

3.7.3 题目大意

法老要用固定数量的石头建造一些金字塔。金字塔分两种：

- 高金字塔　底座为 $n \times n$，上一层为 $n-1 \times n-1$……以此类推。
- 矮金字塔　底座为 $n \times n$，上一层为 $n-2 \times n-2$……以此类推。

给出石块数 n，输出满足以下条件的方案：

1. 所有石块都必须用上；
2. 金字塔数要尽可能少；
3. 所有金字塔两两不同；
4. 金字塔至少包含两层，即底座为 1 的金字塔和底座为 2 的矮金字塔是不允许的；
5. 满足以上 4 点的基础上，最大的金字塔要尽可能大（大的定义为用的石块数多）；
6. 满足以上 5 点的基础上，次大的金字塔要尽可能大；
7. 以此类推…

3.7.4 关键词

背包、搜索

3.7.5 算法讨论

可以看出，金字塔的体积是立方级别的。所以可用的金字塔只有 $O(\sqrt[3]{n})$ 个。

因此，直接背包的复杂度是 $O(n^{\frac{4}{3}})$。7s 的时限下，小心不要让常数过大就能过。

但是，立方级增长远小于指数级增长，在 n 比较大时，物品的分布是比较稠密的。也就是说，n 大于一定限度之后，不会有无解，并且最优方案的期望长度很小。

那么就可以考虑使用迭代搜索。实现后可以发现搜索的效果非常好，并且空间需求减少。

3.7.6 时空复杂度

- 时间：$O(n^{\frac{4}{3}})$[7]
- 空间：$O(n)$

[7]这是背包的复杂度，若用搜索，实际效果快于 $O(n)$.

3.8 2011K: Trash Removal

3.8.1 题目编号

2011 K

3.8.2 题目名称

Trash Removal

3.8.3 题目大意

为了模拟垃圾投入斜道，需要计算在平面上，一个多边形能通过的“斜道”的最小直径。多边形点数 $n \leq 100$。

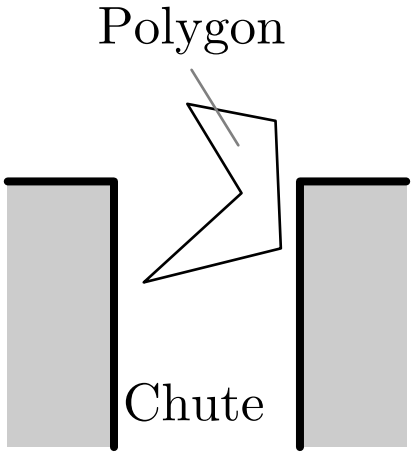

Figure 3: 示意图

3.8.4 关键词

计算几何、暴力

3.8.5 算法讨论

可以证明，最优情况时，至少有三个点在边缘，一个在一边，两个在另一边。

Proof. 明显，最优时至少有两个点在边缘，一个一边。考虑将多边形旋转，让这两个点垂直“斜道壁”的距离减少，直到有其它点碰到“斜道壁”为止。显然，这

不会让直径变少。因此，结论成立。 □

利用上述结论，可以枚举在同一边的两个点，判断是否合法，然后计算直径。判断合法性就是，用叉积判这条直线是否两边都有点，都有则不合法。计算直径：因为两个点确定了一个“斜道壁”的位置，所以直接利用向量运算[8]算出所有点离直线的最大距离即可。

3.8.6 时空复杂度

- 时间：$O(n^3)$
- 空间：$O(n)$

[8]事实上可以利用上一步叉积的结果。

4 ACM/ICPC World Finals 2010

4.1 2010B: Barcodes

4.1.1 题目编号

2010 B

4.1.2 题目名称

Barcodes

4.1.3 题目大意

题目要求对一个条形码进行解码。条形码由深浅交替的宽窄区域表示，且用于解码的只有宽窄信息。用 0 表示窄区域，1 表示宽区域，那么有对应表如下：对于每个合法的条形码，开始和结束都是一个特别的编码；开始码后紧接着原

字符	编码
0	00001
1	10001
2	01001
3	11000
4	00101
5	10100
6	01100
7	00011
8	10010
9	10000
-(10)	00100
开始/结束	00110

字符串的编码，然后两个检验字符 C 和 K，再到结束；每个编码之间用一个 0 隔开。设原字符串有 n 个字符，字符'-' 代表 10，a_i 代表原字符串第 i 位代表的数字；那么计算 C 的方法如下：

$$C = \left(\sum_{i=1}^{n} ((n-i) mod\ 10 + 1) \times a_i \right) mod\ 11$$

同理，另外设 $a_{n+1} = C$，那么有：

$$K = \left(\sum_{i=1}^{n+1} ((n-i+1) mod\ 9 + 1) \times a_i \right) mod\ 11$$

输入扫描到的所有区域的宽度，若合法，那么输出原字符串；若字符 K 错误，输出"bad K"；若字符 C 错误，输出"bad C"；其他情况输出"bad code"。在标准情况下，宽区域的宽度为窄区域的两倍，但是因为印刷问题，最多 $\pm 5\%$ 的误差会被容忍。另外读入的顺序可能为反。原字符串不为空，输入的区域数 ≤ 150，$1 \leq$ 任意区域的宽度 ≤ 250.

4.1.4 关键词

模拟

4.1.5 算法讨论

观察开始/结束的编码，可以发现，是否需要翻转只需要判断第一个编码。如果我们能够知道什么是宽什么是窄，这一题只要判断格式合法性、解码和计算 C 和 K 就可以了。

现在问题在于判断宽窄区域。很明显第一个区域一定是窄的，又因为正确情况下误差只有 $\pm 5\%$，所以随便乱搞就能分开宽窄区域。然后就需要判断误差是否在 $\pm 5\%$ 内。

一个比较简单的方法就是：把窄区域宽度乘 2，然后都变成了宽区域；接着取最小值和最大值，若误差不大于 $\pm 5\%$，那么有最小值比最大值不小于 $\frac{95}{105}$.

4.1.6 时空复杂度

- 时间：$O(n)$
- 空间：$O(n)$

4.2 2010C: Tracking Bio-bots

4.2.1 题目编号

2010 C

4.2.2 题目名称

Tracking Bio-bots

4.2.3 题目大意

在一个平面网格上，物体只能往坐标增大的方向移动。一些条状的网格区域是不能被经过的，这些区域在 y 方向的长度只有 1. 问，在 $(0,0)$ 到 $(n-1,m-1)$ 这 $n \times m$ 个起点中，有多少个可以被经过的，但是无法到达点 $(n-1,m-1)$。$1 \leq n,m \leq 10^6$，条状区域数 $w \leq 1000$。

4.2.4 关键词

暴力

4.2.5 算法讨论

假如 n,m 只有 1000，那么就很好做。设 $f(i,j)$ 表示 (i,j) 是否能走到 $(n-1,m-1)$，即有 $f(i,j) = f(i+1,j)|f(i,j+1)$，然后按顺序转移即可。

但是 n,m 很大……这样，就有两个考虑的方向：

1. 考虑利用条状区域这一个性质。这样，看起来就需要对这些条状区域进行扫描，维护每个扫描到的条状区域下面的网格数、是否能到达终点。并且需要判断起点在条状区域左边的情况。这种做法比较复杂，因为不如下面的思路方便，就不细说了。
2. 因为条状区域最多有 1000 个，所以考虑离散化，然后类比一开始提到的做法去做。需要注意的是，离散化的时候，对于每个条状，x,y 方向要分别取出两个值用于离散。这是因为，对于 x 或 y，离散之后把整条线段分成了三个区域，就需要 2 个界限。

利用上述的第二种方法，就可以很简单暴力的计算出答案。

4.2.6 时空复杂度

- 时间：$O(n^2)$

- 空间：$O(n^2)$

4.3 2010D: Castles

4.3.1 题目编号

2010 D

4.3.2 题目名称

Castles

4.3.3 题目大意

指挥员要攻略一些城堡。一个城堡有三个属性：一是成功攻略所需人数；二是成功攻略后死亡人数；三是成功攻略后要留下来守城的人数。城堡加上之间的道路形成一个树结构。为了某种原因，指挥员要攻略完所有城堡，并每条道路最多经过两次。不过指挥员率领的是天降奇兵，能在任意的城堡开始攻略之旅。求一开始奇兵的最少总人数。城堡数 ≤ 100.

4.3.4 关键词

贪心

4.3.5 算法讨论

首先，死掉的人和留守的人都不能成为战斗力了，所以把这两类合起来。其次，因为每条道路最多经过两次，所以攻略之旅是一次深度优先遍历。因为题目给出的是无根树，所以考虑枚举根。枚举根之后，设 $f(i)$ 表示攻略 i 子树至少需要多少人，$g(i)$ 表示攻略 i 子树会损失多少战斗力。g 值是儿子的 g 值加起来再加上根的消耗，但是 f 值与进入儿子的先后顺序有关。我们拿出所有儿子，排一排；令 a_i 表示排在第 i 位的儿子的 f 值，b_i 是 g 值。同时令 a_0 为根成功攻略所需人数，b_0 为根的消耗数。那么有：

$$f(r) = max\{a_i + \sum_{j=0}^{i-1} b_j\}$$

（解释：加起来的 b 值是之前战斗的消耗，之前的消耗在现在的战斗是用不了的。）这样我们要确定儿子的顺序，让 f 尽量小。考虑两个相邻的儿子 $i, j(i = j-1)$，若交换之后更优，那么有 $max(a_i, a_j + b_i) > max(a_j, a_i + b_j)$. 因为 $a_j + b_i \geq a_j$ 且 $a_i \leq a_i + b_j$，所以需要 $a_j + b_i > a_i + j_j$，也就是 $a_j - b_j > a_i - b_i$. 因此，我们用 a 值减 b 值来排序，大的放前面。每个节点像这样贪心算出 f 值之后，就能取所有有根树的最小值，求出答案。

4.3.6 时空复杂度

- 时间：$O(n^2 \log n)$
- 空间：$O(n)$

4.4 2010F: Contour Mapping

4.4.1 题目编号

2010 F

4.4.2 题目名称

Contour Mapping

4.4.3 题目大意

为了画等高线，测量了若干个点的高度。这些点分为 s 排，奇数排有 p 个点，偶数排有 $p+1$ 个点。以图4为例。其中，约定：

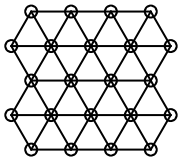

Figure 4: 例：$s=5$、$p=4$

1. 上图的三角形投影到海平面后，每个三角形都是等边三角形，边长为 d。
2. 将每个三角形中的地形视为平整的平面。
3. 等高线画在高度模 H 为 0 的地方，若这种地方同一高度的有连通的一大块区域，那么只在边界画等高线；地图边界也视为边界。

求地图中等高线的长度。$2 \leq s \leq 100$、$1 \leq p \leq 100$、$1 \leq d \leq 10$、$1 \leq H \leq 1000$、$0 \leq$ 点的高度 $\leq 10^6$。

4.4.4 关键词

数学 [9]

4.4.5 算法讨论

因为每个三角形都是平整的，所以考虑一个一个三角形算，然后去重、算边界。

[9]翻译良心的给了三个样例，原题只有两个。

设三角形三个点为 A、B、C，高度不减。那么等高线的两个端点可能在AB和AC 上、或 BC和AC 上。因为等高线在同一种情况中截出的三角形是相似的，所以在同一种情况中等高线的长度是一个等差数列。项数可以除一下算出来，首项通过余弦定理可以算出来，公差利用相似也可算出。另外，为了计算一大块平台的边界的方便，$A = B = C$ 时不计算。

接着就是去重的问题。考虑枚举一条边，若这条边是等高线，且相邻的两个三角形都计算过，那么这条边被重复计算过，去重。

最后是计算一大块平台的边界的问题。考虑上面的计算，若边界在地图内部，那么都已算过，且没有重复。因此，只要计算地图的边界上的长度即可。

4.4.6 时空复杂度

- 时间：$O(s \cdot p)$
- 空间：$O(s \cdot p)$

4.5 2010G: The Islands

4.5.1 题目编号

2010 G

4.5.2 题目名称

The Islands

4.5.3 题目大意

平面上有一些点，x 坐标均不相同。要求一个简单环，从最左边的点出发，经过的点 x 坐标单调递增，到最右边的点，再按 x 坐标单调递减的顺序经过剩下所有点回到起点。

另外，有两个在中间的点，不能同时出现在从左到右或者从右到左的过程中。

求环的最小长度，并输出方案。数据保证唯一解。

点数 $n \leq 100$。

4.5.4 关键词

动态规划

4.5.5 算法讨论

因为处理环比较麻烦，所以考虑分成两个除了起点终点不交的单调路径。

那么，就可以用 $f(i,j,s)$ 表示，当前路径 1 末端为 i，路径 2 末端为 j，两路径含有特殊点的情况为 s，下的最小长度。如此，忽略 s 的转移就是

$$f(i,j) = \begin{cases} f(i-1,j) + dis(i,i-1) & i > j+1 \\ \min\{f(k,j) + dis(i,k) | k < j\} & i = j+1 \\ \min\{f(i,k) + dis(j,k) | k < i\} & i+1 = j \\ f(i,j-1) + dis(j,j-1) & i+1 < j \end{cases}$$

当然，$f(n,n)$ 的转移特殊一点，但是本质与上面的转移相同。明显，转移的均摊复杂度是 $O(1)$ 的。

加上 s 之后，除了维护 s 的值，只是当 $\max\{i,j\}$ 是特殊点时，根据 s 的值，有一些转移不可行。转移方程本质上是一样的。

这样，DP 时记录转移的父亲结点，就可以在最后还原出两条路径，再处理一下可以得到方案。

4.5.6 时空复杂度

- 时间：$O(n^2)$
- 空间：$O(n^2)$

4.6 2010H: Rain

4.6.1 题目编号

2010 H

4.6.2 题目名称

Rain

4.6.3 题目大意

有一片丘陵地区，需要预测在最坏情况下，暴雨形成的堰塞湖的状况。

题目给出一个三角划分模拟地形：输入每个测量点的高度，互相连边的情况。丘陵地区外的高度都低于丘陵地区的高度。

输出最坏情况下，形成的每个湖湖面的海拔高度。

测量点数 $n \leq 2704$。

4.6.4 关键词

平面图、最短路

4.6.5 算法讨论

一个显然的性质是，*一个湖包含不少于一个三角*。[10]

因为三角划分是一个*平面图*，所以考虑把平面图的面处理出来。[11]

处理出所有面之后，枚举一下即可找到每个面的最低高度 $low(F_i)$ 和边的最低高度 $low(E_i)$。

然后，从所有外部面出发，做“最短路”。设 $f(F_i)$ 表示面 i 最高的水面高度[12]；在平面图的对偶图上转移，一个面到另一个面时，f 值 max 上转移边对应边的最低高度、目标面的最低高度。

这个算法是显然正确的。用这个算法算出 f 之后，相邻的 $low(E_i) < f(E_i)$，并且相隔的边最小的最低高度小于 $f(E_i)$ 的面会*连成一片湖*。

[10]因为一个三角形代表的是一个斜面。

[11] 一个比较简单的算法是：把一条边拆成两条有向边；对每个点，排序从它出发的所有有向边；任选一个没遍历过的边出发；每一步先找到当前边的反向边，再找到反向边按逆时针顺序的下一条边，并把当前边变成它；这样，每一次遍历就可以找出一个面；内部面是顺时针顺序遍历边的，外部面是逆时针顺序遍历的。

[12]当 $f(F_i) == low(F_i)$ 时，相当于没有积水。

这样，就可以用 floodfill 找出每一片湖。最后将湖面海拔排一下序输出即可。

这个算法看上去并不难，却由很多并不很短的子算法组成，实际上编程复杂度不低。

但是，上面的做法没有使用到题目提供的平面图是三角划分这个性质，或许利用这个性质改进算法可以让程序更短。[13]

4.6.6 时空复杂度

- 时间：$O(n^2)$[14]
- 空间：$O(n)$

[13] 某第一次写的时候，利用面都是三角形这个性质来判断一个点是否在外部面。接着 assert 外部面点数大于零。然后……交到 LA 上 RE 了（去掉 assert 之后 WA）。最后重写换成了上述算法。

[14] 预处理一条边的反向边的位置可以做到 $O(n \log n)$。

4.7 2010I: Robots on Ice

4.7.1 题目编号

2010 I

4.7.2 题目名称

Robots on Ice

4.7.3 题目大意

给出一个 $n \times m$ 的网格，四连通，满足另一些条件下，求一条起点为 $(0,0)$，终点为 $(0,1)$ 的哈密顿路径。同时，输入给定第 $\lfloor \frac{inm}{4} \rfloor$ 步时，所在的位置。

$2 \leq n, m \leq 8$。

4.7.4 关键词

搜索

4.7.5 算法讨论

因为网格最大 8×8，所以两个限制之间最多差 16 步。又因为限制太严，很多算法都不适用，所以考虑搜索。

虽然题目的限制比较严，但是对于直接搜索还是太松了，因此需要增加剪枝。

- 一个明显的剪枝：限制点不能提前到达。
- 因为一个格子到另一个格子至少需要曼哈顿距离步，所以判断剩下的时间和下一个限制点之间的曼哈顿距离来剪枝。
- 因为是哈密顿路径，所以不能把剩下未走完的分成两半。利用这个性质就可以增加一个剪枝：每扩展一个点，利用周围 4 个点，判断是否分割了剩下的部分。
- 另外，可以将终点 $(0,1)$ 视为一个限制点。

4.7.6 时空复杂度

- 时间：$O(?)$
- 空间：$O(n^2)$

4.8 2010J: Sharing Chocolate

4.8.1 题目编号

2010 J

4.8.2 题目名称

Sharing Chocolate

4.8.3 题目大意

题目要求将一个矩形分成若干个指定面积的矩形。分出来的矩形各自的面积给定，而且每次分割时要一刀把一个矩形断成两个矩形，只允许刀沿直线切下去。求是否可行。分出的矩形数 ≤ 15，大矩形的长 ≤ 100.

4.8.4 关键词

状压 DP

4.8.5 算法讨论

因为矩形数很少，所以考虑状压 DP。用 $f(s,i,j)$ 表示集合 s 里的小矩形是否能组成一个 $i \times j$ 的矩形。然后可以套上记忆化搜索，枚举横着还是竖着割开矩形、哪边有什么小矩形。因为集合确定了之后面积确定了，所以 $f(s,i,j)$ 在实际存储的时候可以省去 i 或 j. 主要的时间在于枚举一个二进制的子集，这一步是可以做到均摊 $O(3^n)$ 的，所以时间是足够的。事实上 $O(4^n)$ 常数小也能过——面积的剪枝让实际复杂度远小于理论复杂度。

4.8.6 时空复杂度

- 时间：$O(L \cdot 3^n)$
- 空间：$O(L \cdot 2^n)$

4.9 2010K: Paperweight

4.9.1 题目编号

2010 K

4.9.2 题目名称

Paperweight

4.9.3 题目大意

有一个纸镇，由两个四面体组成。两个四面体的一个面完全重合。在这个纸镇表面上、或里面，有一个特殊点。一个摆放的物体“稳定”，就是说，重心向任何方向移动至多 0.2 距离，物体都不会有滚动的趋势。同时，纸镇的密度均匀。要求将纸镇摆放在平面上，使得特殊点离平面最近或最远，而且纸镇“稳定”。求这两种情况下，特殊点离平面的距离。坐标绝对值 ≤ 1000.

样例纸镇如图5，红点为特殊点。

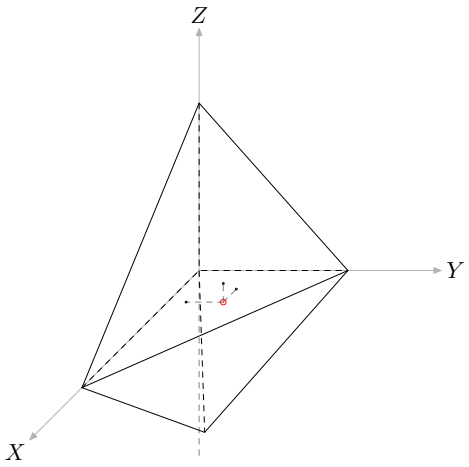

Figure 5: 样例

4.9.4 关键词

计算几何

4.9.5 算法讨论

首先，考虑计算重心的位置。因为纸镇的密度均匀，所以重心等于几何中心。单独一个四面体的几何中心是很好求的，就是四个点的平均值。以样例为例，图6中，绿点为两个四面体分别的重心，蓝点为纸镇的重心。明显纸镇的重心必在另两个重心的连线上[15]。因为重心是质点，所以这里就可以运用杠杆原理[16]求出重心。

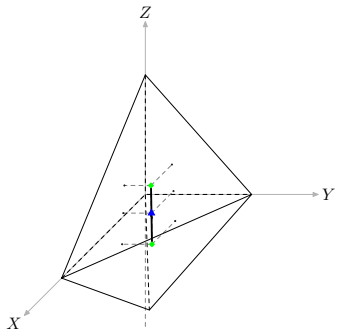

Figure 6: 三重心

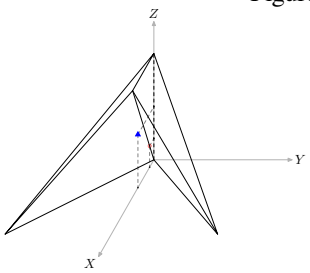

Figure 7: 情况一

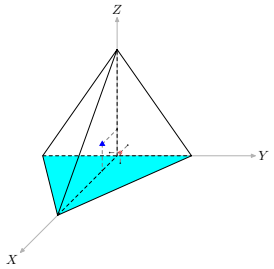

Figure 8: 情况二

求出重心之后，为了求答案，需要枚举平面。考虑枚举三个顶点确定一个平面；图7说明了支点并不一定在四面体的一个面上[17]。因为除了判断所有点是否在平面的一边，还要判断重心的情况。所以，就需要计算支面，及重心在支面上的位置。考虑用在枚举平面上的点做一个平面凸包；图8说明了支面并不一定是三角形[18]。这样就算出了支面。

对于平面凸包，可取两个不共线向量作 X、Y 轴，坐标是分别的点积，就映射到平面上了。判断重心的位置就需要得到，在平面上，垂直于凸包边，向凸包内的向量。这可以通过解

$$(t\overrightarrow{i}+\overrightarrow{j})\cdot\overrightarrow{i}=0$$

[15]因为沿连线切一刀，两个四面体都被平分了。
[16]其实就是加权平均数…
[17]图中蓝色三角为中心，红点为特殊点。下同。
[18]支面为青色区域，其他如上。

得出 t 来得到一个符合的向量

$$t\vec{i} + \vec{j}$$

其他的计算都是简单的三维向量运算，略。

4.9.6 时间复杂度

因为点数是固定的，所以对于一个纸镇来说计算的时间是 $O(1)$ 的。

- 时间：$O(1)$
- 空间：$O(1)$

5 ACM/ICPC World Finals 2009

5.1 2009A: A Careful Approach

5.1.1 题目编号

2009 A

5.1.2 题目名称

A Careful Approach

5.1.3 题目大意

给出一些区间，对于每个区间，在这个区间范围内选一个实数，使得任意两个实数之间最小的差最大。区间数不大于 8。所有区间属于 $[0, 1440]$.

5.1.4 关键词

二分、贪心、暴力

5.1.5 算法讨论

给出区间选数似乎有点难以从正面入手。

假如从各区间选出来的数的大小顺序确定、答案确定，那么就能够使用贪心在有解情况下构造一个解。构造方法：从小到大选数，选出来的数尽量小（为上一个数加答案和这个区间的最小值这两个数的最小值）。显然这样能确保有解时能构造出可行解。

利用上述的贪心，我们就可以得出解决这道题目的算法。首先二分答案，然后搜索数的大小顺序，再采用上述贪心判断解的存在性。当然，第一步和第二步可以互换。

5.1.6 时空复杂度

- 时间：$O(n! \cdot n \log C)$
- 空间：$O(n)$

5.2 2009B: My Bad

5.2.1 题目编号

2009 B

5.2.2 题目名称

My Bad

5.2.3 题目大意

给出一个无环的逻辑电路。逻辑门有四种：与门、或门、异或门、非门。

其中有至多一个门坏了，有三种坏的方式：

1. 总是输出 0.
2. 总是输出 1.
3. 输出总是错误。

给出若干组测试的输入、输出；每组测试的输入互不相同。判断电路的状况。

电路输入数 $n \leq 8$，门数 $G \leq 19$，电路输出数 $U \leq 19$。

5.2.4 关键词

枚举

5.2.5 算法讨论

因为电路无环，所以考虑拓扑排序；得到输入之后扫一遍处理一下就可以得到输出。

这样，每次计算输出的复杂度是 $O(G)$。

判断电路是否有损坏之后，因为只有一个门损坏，所以考虑枚举门和损坏的方式，然后代入所有测试判断电路输出是否相符。

最后，就可以根据统计得出的结果判断电路的状态。

5.2.6 时空复杂度

- 时间：$O(2^n \cdot G^2)$

- 空间：$O(2^n \cdot G)$

5.3 2009D: Conduit Packing

5.3.1 题目编号

2009 D

5.3.2 题目名称

Conduit Packing

5.3.3 题目大意

给出四个圆的直径，求一个最小的直径，使得这个直径的圆能让这四个圆互不相交、互不包含地放在里面。

5.3.4 关键词

计算几何、二分搜索

5.3.5 算法讨论

考虑二分答案。观察最优放置情况：

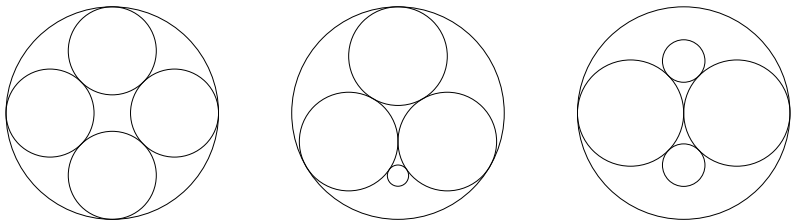

Figure 9: 三种情况

可以发现，一个圆至少与两个圆相切。

那么考虑搜索放置方案；每一步搜索，枚举当前放哪个圆，与哪两个圆相切，圆心在两个解中的哪个。显然，第一个放入的圆当时只与大圆相切，不需要枚举两个圆。

因为只有 4 个圆需要搜索，所以实际复杂度不大。

求圆心位置可以转化为求两圆交点，利用式(2)即可算出两圆交点。

5.3.6 时空复杂度

- 时间：$O(?)$
- 空间：$O(1)$

5.4 2009E: Fare and Balanced

5.4.1 题目编号

2009 E

5.4.2 题目名称

Fare and Balanced

5.4.3 题目大意

给出一个 n 个点的有向无环图，要求增加一些边的费用，使得：

- 1 到 n 的距离尽可能小。
- 每条 1 到 n 的路线，长度都一样。
- 每条 1 到 n 的路线，至多只有一条边被增加费用。

输出任意方案。

$n, m \leq 5 \times 10^4$。

5.4.4 关键词

动态规划、拓扑序

5.4.5 算法讨论

- 设 $h(i, j)$ 表示 i 到 j 是否有多种不同长度的路线。
- 那么若存在 $h(1, i) = true$，且 $h(i, n) = true$，就一定无解。
- 根据 $h(i, j)$ 的定义，还有 $h(1, i)$ 和 $h(i, n)$ 不同时为真，那么，对于每条长度小于最长路的线路，存在 $(u, v) \in E, h(1, u) = false, (1, v) = true$，边 (u, v) 只属于这条路线。

  Proof. 因为有长度小于最长路的线路，所以 $h(1, n) = true$；又根据 $h(1, i)$ 和 $h(i, n)$ 不同时为真，那么存在 u, v 在路线上，使得 $h(1, v) = true, h(v, n) = false, h(1, u) = false, h(u, n) = true$。因为 $h(1, u) = false, h(v, n) = false$，所以边 (u, v) 只存在于“一”条线路上——相同长度的线路相当于一条。 □

- 那么为了计算方案，就要知道 1 到 i 和 i 到 n 的最长路；每条符合上述要求的边 (u, v) 都要增加 $dis(1, n) - dis(1, u) - dis(v, n) - cost(u, v)$。计算最长路和 h 只要用 DP 和拓扑序即可。

5.4.6 时空复杂度

- 时间：$O(n + m)$
- 空间：$O(n + m)$

5.5 2009F: Deer-Proof Fence

5.5.1 题目编号

2009 F

5.5.2 题目名称

Deer-Proof Fence

5.5.3 题目大意

有一些点，需要用若干个封闭的围栏将这些点围起来。对于任意一个点，围住它的围栏离它的最小距离不小于 m。求围栏的最小总长度。点数 $n \leq 9$，$m \leq 200$，坐标绝对值大小 ≤ 100。

5.5.4 关键词

计算几何

5.5.5 算法讨论

因为 n 很小，考虑枚举点的集合计算所需围栏长度；然后 DP 合并。

对于一个点集，围栏明显是它们的凸包的外扩；并且，在每个角，用圆弧*圆滑地*包住。可以发现，围栏的长度就是凸包周长加上半径为 m 的圆的周长。

计算得出围栏长度之后，通过计算 $f(S) = \min\{f(i) + f(i \oplus S)\} | i \in S$，就可得出答案。

5.5.6 时空复杂度

- 时间：$O(2^n \cdot n^2 + 3^n)$
- 空间：$O(2^n)$

5.6 2009G: House of Cards

5.6.1 题目编号

2009 G

5.6.2 题目名称

House of Cards

5.6.3 题目大意

Axel 和 Birgit 玩这样的一种纸牌游戏：他们建造一个由纸牌组成的房子，当他们添加纸牌到房子的时候，会获得（或失去）游戏的分数。Axel 和 Birgit 只使用 2 种花色，1 红 1 黑。每种花色有 13 个等级。我们使用记号 **1R**, **2R**, . . . , **13R**, **1B**, **2B**, . . . , **13B** 来表示等级和颜色。

开始前，玩家要选择半副标准纸牌的一个子集，子集中所有纸牌的最大等级是 M。洗完选出的纸牌后，他们从牌堆的最上面拿出 8 张，从左到右连续地放置它们形成 4 个“山峰”。举个例子，如果 $M = 13$ 而且前 10 张纸牌（26 张的前 10 张）是：

6B 3R 5B 2B 1B 5R 13R 7B 11R 1R . . .

那么这个游戏开始的时候就像图10所展示的那样。

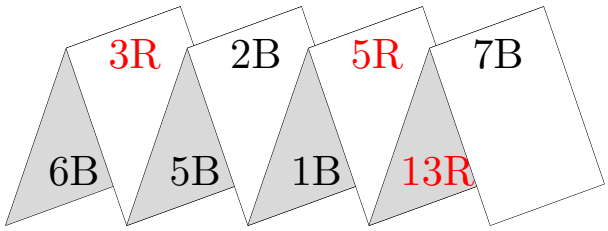

Figure 10: 初始“山峰”示例

每个玩家被认定一种颜色，Birgit 总被认定是黑色，Axel 总被认定是红色。第一张用于组成山峰和山谷的纸牌的颜色决定了哪个玩家先开始。图10的那个例子，Birgit 先开始，因为第一张纸牌是 6B。

玩家交替进行操作。一步操作包括从一排纸牌的最前面抽取一张纸牌然后进行下列的一条：

1. 持有这张纸牌直到下次操作（这是一张“被持有的纸牌”），任何时候最多持有 1 张纸牌。

2. 用刚抽取的纸牌或被持有的纸牌覆盖在两个山峰之间的山谷，形成一个“基底”。如果还剩下一张牌，那么这张牌就被持有。
3. 把 2 张纸牌放在基底上面，形成一个山峰（其中一张纸牌一定是一张被“持有”的纸牌）。

重要的是，两个玩家在纸牌被抽取前就事先知道纸牌的顺序。

如果玩家通过添加了一个基底组成了一个向下的三角形，或者通过添加了一个山峰组成了一个向上的三角形，那么玩家的分数就会像下面描述的那样更新。组成三角形的 3 张纸牌的等级之和将被增加到那个颜色与 3 张纸牌的多数颜色相等的那个玩家的分数上。如果在游戏中没有组成三角形，两个玩家的分数保持不变。

如果在某步操作结束后没有纸牌等待被抽取，这个游戏就结束了。如果某个玩家在这个时候持有纸牌，那个玩家的分数将会增加（或减少）这张纸牌的等级如果这张纸牌的颜色与玩家颜色相同（不同）。

求两个玩家都采取最优策略时的最终结果。$5 \leq M \leq 13$。

5.6.4 关键词

博弈搜索

5.6.5 算法讨论

一个有趣的事实是，当 $M = 13$ 时，用尽所有纸牌可以造出一个完整的纸塔，如图11。

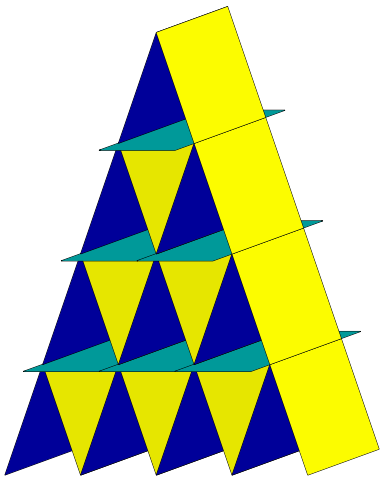

Figure 11: 完整的纸塔

因为 M 不是很大，所以考虑博弈搜索加上 alpha-beta 剪枝[19]。在实践中，加上 alpha-beta 剪枝就足够快出解了。

这题的难点在于纸塔状态的处理。我们可以类比正常的搜索，用二维数组表示竖着、横着的纸牌位置的纸牌状态，搜入下一层时改动数组，搜完即恢复。这样写就好处理多了，虽然某写了 5k+…其他的就是各种情况的处理，小心注意即可。

5.6.6 时空复杂度

- 时间：$\ll O(6^{2M-8})$
- 空间：$O(M^2)$

[19]因为 alpha-beta 剪枝很简单，所以参见维基。

5.7 2009H: The Ministers' Major Mess

5.7.1 题目编号

2009 H

5.7.2 题目名称

The Ministers' Major Mess

5.7.3 题目大意

有一些议案，大臣每人最多向 4 个议案表决支持或否定。一个合法方案即满足每个大臣超过一半的表决符合结果。问是否存在方案，若存在，问有哪些议案的结果必须是什么。议案数 ≤ 100，大臣数 ≤ 500.

5.7.4 关键词

图论

5.7.5 算法讨论

因为每个大臣超过一半的表决符合结果，所以大臣有两类：

表决一次或两次 *表决都要符合结果。*

表决三次或四次 *表决最多有一个不符合结果。*

这样，考虑构造一个图，将每个议案分为两个点，通过和不通过。对于第二类的大臣，枚举哪个议案不符合，然后从不符合的点向符合他其他表决的点连有向边，表示依赖关系。从每个点出发，根据依赖关系遍历点，若出现一个议案两个点都依赖或依赖了必不选的点，则出发点不可选。然后处理第一类大臣；第一类里的点是必选的，接着根据上一步的结果判断方案的存在性，最后就可以根据哪些点可选来输出答案。

5.7.6 时空复杂度

- 时间：$O(n \cdot m)$
- 空间：$O(n + m)$

5.8 2009I: Struts and Springs

5.8.1 题目编号

2009 I

5.8.2 题目名称

Struts and Springs

5.8.3 题目大意

平面上有一些矩形。这些矩形都是嵌套的，没有相交。所有小矩形都被一个大矩形所包含。小矩形内有垂直和水平两条支柱／弹簧；小矩形与直接包含它的矩形有垂直上下两条支柱／弹簧，和水平左右两条支柱／弹簧。支柱的长度不能变化；一个矩形的至多 6 条弹簧，在相同方向上的，长度变化的比例要相同。有多次操作，把大矩形的长宽变化，求里面的矩形的坐标和长宽。小矩形数 ≤ 500，询问数 ≤ 500，大矩形边的大小 $\leq 10^6$.

5.8.4 关键词

树结构、模拟

5.8.5 算法讨论

经过观察，可以发现嵌套结构就是树结构。

将矩形按照宽度排序，因为小矩形包含不了大矩形，所以这是一个拓扑序。因此，找矩形的父亲时可以直接倒着找；并且处理询问时扫一遍处理即可。

这题处理询问的关键是算出每条弹簧的变化。由父亲矩形的大小变化可以得知每个方向上总的变化，然后要根据每条弹簧的长度成正比分配变化。

在这里值得注意的是，因为是整数运算，乘起来再除的时候，乘法结果可能超过 32 位整数。

5.8.6 时空复杂度

- 时间：$O(n^2 + n \cdot Q)$
- 空间：$O(n)$

5.9 2009J: Subway Timing

5.9.1 题目编号

2009 J

5.9.2 题目名称

Subway Timing

5.9.3 题目大意

在一棵树上，每条边的经过所需时间单位为秒。现在要将边的时间上取整或下取整化为分钟。要求，取整后任两个结点之间所需时间的误差的最大值最小。

点数 $n \leq 100$。

5.9.4 关键词

二分、贪心

5.9.5 算法讨论

可以证明，答案不会超过 $120 = 60 \times 2$。[20]

Proof. 在树上任选一个根，使得每个结点都有父亲。对于每个点，它与父亲的连边选择任一种可行的取整方式，保证它到根的路径的误差在 $[-30, 30]$ 这个区间内。

这个选择方式是可行的：选择之后，到根的路径误差为 a 或 b（假设 $a < b$），那么 $b - a = 60$；又因为父亲到根路径的误差 $x \in [-30, 30]$ 且 $a \leq x \leq b$，所以 a 和 b 必至少有一个落在区间 $[-30, 30]$ 内。

那么，一个结点到它任一祖先的路径的误差的绝对值不大于 60；因为可以将任意路径分解成最多两条结点到祖先的路径，所以任意路径误差的绝对值不大于 120.[21] □

于是，二分答案之后，就可以采用一个依赖答案大小的贪心。

- 当二分的答案是 K，对于结点 u，若误差区间 $[a, b]$ 是可接受的，就满足：
  - 子树任意路径的误差的绝对值不大于 K。

[20] 因此答案约与 n 同阶。
[21] 更进一步的，可以证明答案不大于 118，但是对于本题来说已经差不多了。

  - 子树任意结点到根路径的误差在区间 $[a, b]$ 内。

- 对于结点 u 的可接受误差区间 $[a, b]$，若不存在可接受区间 $[a, b']$，使得 $b' < b$，那么区间 $[a, b]$ 是精简的。
- 若当前在计算结点 u 的精简可行区间，那么设 T_i 表示计算了前 i 个儿子合并的结果。
- 利用 T_{i-1} 和第 i 个儿子的结果计算 T_i：
  1. 枚举 T_{i-1} 的区间 $[a, b]$，第 i 个儿子的区间 $[a', b']$，和第 i 个儿子连向父亲的边的误差 e；
  2. 若枚举结果能合并，那么满足：$a + a' + e$ 和 $b + b' + e$ 的绝对值的最大值不大于 K。
  3. 枚举的结果合并之后，得到可行区间 $[\min\{a, a' + e\}, \max\{b, b' + e\}]$。
- 要注意的是，合并得出的区间不一定精简，所以需要把非精简区间去掉；不然每个结点的区间数就从 $O(n)$ 上升为 $O(n^2)$ 了。
- 判断答案是否合法，只要判断根是否有可行区间即可。

5.9.6 时空复杂度

- 时间：$O(n^3 \log n)$
- 空间：$O(n^2)$

6 ACM/ICPC World Finals 2008

6.1 2008A: Air Conditioning Machinery

6.1.1 题目编号

2008 A

6.1.2 题目名称

Air Conditioning Machinery

6.1.3 题目大意

有一个长方体空间，要求用至多 6 个 L 型的弯管连接两个进出口。如示意图。

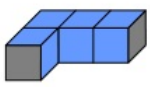

Figure 1

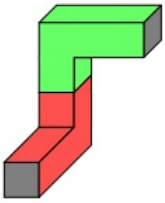

Figure 2

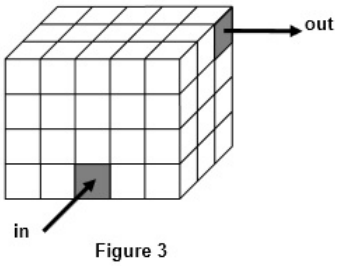

Figure 3

长方体的大小不大于 $20 \times 20 \times 20$.

6.1.4 关键词

搜索

6.1.5 算法讨论

因为弯管只有六个，所以考虑迭代搜索。从一个口出发，有 8 种放 L 型弯管的方案，因此搜索的规模在 $8^6 = 262144$ 左右，是足够的。

在实现方面，可以考虑一个一个方格地搜索，记录现在是哪个弯管，弯管的哪个位置。这样，搜索过程的编程的复杂度会比较小。

6.1.6 时空复杂度

- 时间：$O(8^n)$
- 空间：$O(L^3)$

6.2 2008B: Always an Integer

6.2.1 题目编号

2008 B

6.2.2 题目名称

Always an Integer

6.2.3 题目大意

给出一个整系数多项式 P，问代入正整数时，结果能否被 D 整除。

P 的次数 ≤ 100。

6.2.4 关键词

字符串处理、数学、递归

6.2.5 算法讨论

首先，这题的输入类似`(-n^14-11n+1)/3`，需要处理一下，转成方便计算机使用的形式[22]。

正面解决这个问题似乎很难，因此考虑减小问题的规模。若一个集合的所有数模 D 为零，首先其中一个模 D 为零，其次，任意元素与之的差模 D 也为零。那么就可以转化成两个子问题：

- 检验 $P(1)$ 模 D 为零。
- 检验对于 $\forall n \in Z^+$，$P(n+1) - P(n)$ 模 D 均为零。

明显，第一个子问题线性时间即可解决；第二个子问题就是原问题的缩小版，最高次数减少了一。

于是，就可以不停的将问题规模减少，直到最高次数为零，就可以直接得出结果了。

但是，这又有另外一个问题：问题规模减少的同时，多项式系数的增长也很快。不过，因为问题是判断模 D 是否为零，所以可以将系数全模 D，不影响最终结果。

[22] C++ 的 $atoi$ 就派上用场了。

6.2.6 时空复杂度

- 时间：$O(n^3)$
- 空间：$O(n^2)$

6.3 2008E: Huffman Codes

6.3.1 题目编号

2008 E

6.3.2 题目名称

Huffman Codes

6.3.3 题目大意

给出一些哈夫曼代码，求有多少可能的字频分布可以得出这些哈夫曼代码。

为了简化问题，

- 合并两个哈夫曼树时，权值较小的为左儿子；
- 分配哈夫曼代码时，连向左儿子的边为零，连向右儿子的边为一；
- 每个字母的频率为百分之一的整数倍。

代码个数 $n \leq 20$。

6.3.4 关键词

搜索

6.3.5 算法讨论

因为 n 比较小，而且这题模型不容易转化，所以考虑搜索加剪枝。

- 考虑将合并哈夫曼树的过程逆过来。那么，就有分裂操作。可以看出，分裂操作分出来的两个权值，任何一个均不大于之前分裂操作产生的最小权值。*证明：考虑反过来，合并哈夫曼树，找到两个最小的权值；这样，先分裂产生的就是后合并的，所以先分裂出的权值不小于后面分裂出的权值。*
- 由上一个规律，可以发现，每一次分裂都是分裂当前最大的权值。*证明：若先分裂了小于最大值的权值，那么当后面分裂最大值的时候，无论怎么分都会有至少一个权值大于这次分裂出的权值之一。*

然后，利用这两个规律搜索、剪枝，就可以很快出解了。

6.3.6 时空复杂度

- 时间：$O(?)$
- 空间：$O(n^2)$

6.4 2008F: Glenbow Museum

6.4.1 题目编号

2008 F

6.4.2 题目名称

Glenbow Museum

6.4.3 题目大意

若一个简单多边形的角都是 90° 或 270° 的，那么用 R 代表 90° 的角，用 O 代表 270° 的角，按逆时针顺序可以得到一个字符串。题目称这个字符串为角序列。定义一个角序列合法，即存在一个简单多边形对应这个角序列，并且在这个多边形内有一点可以看到所有的周长。求给定长度的合法角序列的数量。长度 ≤ 1000.

6.4.4 关键词

组合数学

6.4.5 算法讨论

经过观察可以发现，合法角序列中不能有连续的 O，R 的数量比 O 多 4. 这个条件是充要的。

考虑一个凸出的疙瘩，它的角序列明显是 O...O。假如一个疙瘩后有一个 R，那么下一个疙瘩就会换一个方向。又因为若要不合法，明显只能让一个方向有多于一个的疙瘩。所以没有连续的 O 是能构成多边形的角序列合法的充分条件。必要性显然。这样，问题就转化为计算一个 0 和 1 数量给定，没有连续的 0 的序列的个数。

考虑先删去所有 O，那么剩下 $\frac{n+4}{2}$ 个 R。然后选 $\frac{n-4}{2}$ 个 R，把 O 插入到这些 R 的后面。令 $m = \frac{n+4}{2}$，这样就有 $C(m-4, m) = C(4, m)$ 个方案。但是这些方案的第一个字符是 R，所以还要把 O 排在第一位再算一遍：$C(4, m-1)$. 因此最后的答案是 $C(4, m) + C(4, m-1)$。当然，长度为奇数的时候没有能构造多边形的角序列。

6.4.6 时空复杂度

- 时间：$O(Q)$

- 空间：$O(1)$

6.5 2008G: Net Loss

6.5.1 题目编号

2008 G

6.5.2 题目名称

Net Loss

6.5.3 题目大意

题目要求算出两段线段来拟合一个多项式函数。两条线段在 $x = c$ 处相交 $(-1 < c < 1)$。函数的定义域为 $[-1, 1]$。给出多项式函数 $p(x)$，两条线段：

$$g(x) = \begin{cases} a_0 \cdot x + a_1 & -1 \le x \le c \\ b_0 \cdot x + b_1 & c \le x \le 1 \end{cases}$$

函数 $p(x)$ 和 $g(x)$ 的方差为：

$$\int_{-1}^{1} [p(x) - g(x)]^2 dx$$

两个函数方差越小，就拟合得越好。另外，$p(x)$ 的次数 n 不大于 10 。现在 c 已经给定，求 a_0, a_1, b_0, b_1.

6.5.4 关键词

数学

6.5.5 算法讨论

设两线段交点在 (c, d)。将 $\int_{-1}^{c} [p(x) - g(x)]^2 dx$ 展开，可以得到

$$\begin{aligned} &\int_{-1}^{c} p(x)^2 dx - 2a_0 \cdot \int_{-1}^{c} x \cdot p(x) dx - 2a_1 \cdot \int_{-1}^{c} p(x) dx \\ &+ \frac{a_0^2 \cdot (c^3 + 1)}{3} + a_0 a_1 (c^2 - 1) + a_1^2 \cdot (c + 1) \end{aligned} \tag{1}$$

那么代入 $a_1 = d - c \cdot a_0$ 到(1)，就能得到一个更长的、关于 a_0 和 d 的式子。将 d 视为常数，那么这就是一个关于 a_0 的二次函数。又因为 $y = ax^2 + bx + c$ 的最小值为 $c - \frac{b^2}{4a}$，所以取最小值后 a_0 被去掉。这样就剩下了一个关于 d 的二次函数。

同理，对 b_0, b_1 计算出关于 d 的另外一个二次函数，加起来就可以求出 d 值。算出 d 之后代入到 $a_1 + c \cdot a_0 = d$ 再与(1)联立，即可求出 a_0, a_1。同理可求得 b_0, b_1。

对于上述二次函数的二次项，我们最终可以发现，都是 c 的三次函数。画出图像之后，零点为 -1 或 1. 因为 $-1 < c < 1$，所以上面的二次函数二次项不为零，不需要分类讨论。

6.5.6 时空复杂度

复杂度在于多项式乘法。

- 时间：$O(n^2)$
- 空间：$O(n^2)$

6.6 2008H: Painter

6.6.1 题目编号

2008 H

6.6.2 题目名称

Painter

6.6.3 题目大意

题目要求对一些三角形，计算包含关系的最大深度。若这些三角形之间有相交，或公共点，那么输出信息并终止这组数据的计算。三角形都没有退化为线，三角形个数 $n \leq 10^5$，坐标均为整数且坐标绝对值 $\leq 10^6$。

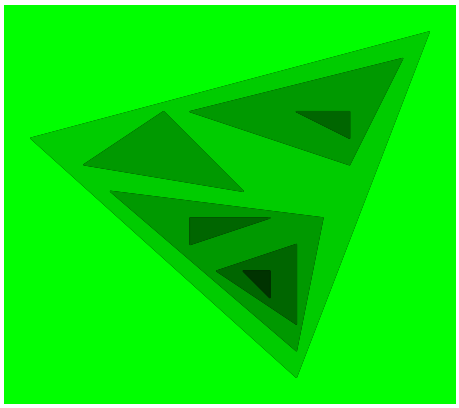

Figure 12: 样例：5 种颜色的三角形单色画

6.6.4 关键词

计算几何、扫描线

6.6.5 算法讨论

这道题明显由两部分组成。

判断三角形相交： 将每个三角形拆成三条线段，如此就可以上判断是否有线段相交的算法——这是有标准算法的；另外这题中线段相交时，要判断

是否在同一三角形内。判线段相交，可以采用扫描线，维护与扫描线相交的线段在一个方向上的顺序；若严格相交，那么相交的两条线段的顺序必定会变化。在端点相交时，需要特判；并且平行于扫描线的线段也需要特殊处理——另外，一种避免这种囧况的方法就是把坐标系随机旋转一个奇怪的角度。

计算包含关系： 计算包含关系也可采用扫描线。标记线段属于三角形的上半部分还是下半部分。每次扫到一个上部分的线段 a，找到扫描线上，第一个高于它的线段 b；如果线段 b 属于上半部分，那么 a 所属的三角形直接被 b 所属的三角形包含；如果属于下半部分，那么 a 所属的三角形与 b 所属的三角形是兄弟，被同一个三角形直接包含。这样就可以处理出一棵树，答案就是树的最大深度。

因为上述两部分都采用的扫描线，所以实际上可以直接合并在一起运算。

6.6.6 时空复杂度

- 时间：$O(n \log n)$
- 空间：$O(n)$

6.7 2008I: Password Suspects

6.7.1 题目编号

2008 I

6.7.2 题目名称

Password Suspects

6.7.3 题目大意

要求算出符合要求的字符串个数。要求：

- 长度为 n。
- 给出 m 个子串，均要包含。
- 每个字母为'a'-'z' 。

若字符串个数不大于 42，则按字典序输出全部合法字符串。

$1 \leq n \leq 25$，$0 \leq m \leq 10$，子串长度 ≤ 10。

6.7.4 关键词

AC 自动机、动态规划

6.7.5 算法讨论

因为要包含所有给定子串，所以考虑在 AC 自动机上 DP。

- 先利用给定子串构造一个 AC 自动机。
- 对于自动机上的结点，在 BFS 计算 fail 指针时，顺便计算结点 i *对应的字符串及其任意后缀*组成的集合中，存在哪些给定子串 S_i。
- 那么，就可以设状态 $f(k, i, j)$ 表示，DP 当前的字符串长度为 k，在自动机的结点 i 上，已包含子串的集合为 j 时，的方案数。
- 然后 f 就可以在自动机上转移：枚举下一个字母；找到到达的结点 p，集合 j 并上集合 S_p，就是转移到的状态。
- 如此，方案数就是 $\sum f(n, i, 2^m - 1)$。

- 当需要输出方案时，考虑再写一个 DP；用 $g(k,i,j)$ 表示状态 $f(k,i,j)$ 最后是否能转移到 $f(n,i,2^m-1)$ 之一。那么转移就是 $f(k,i,j)$ 的转移倒过来。
- 计算完 $g(k,i,j)$ 之后，就可以用一个 DFS，枚举每一位的字母，并在自动机上走；然后，利用 $g(k,i,j)$ 剪枝即可快速出解。

6.7.6 时空复杂度

- 时间：$O(nm \cdot 2^m)$
- 空间：$O(nm \cdot 2^m)$

6.8 2008J: The Sky is the Limit

6.8.1 题目编号

2008 J

6.8.2 题目名称

The Sky is the Limit

6.8.3 题目大意

题目要求计算天际线的长度。在地平线上排列着一些山，都是一个个等腰三角形，底在地平线上。山之间可能有重叠。给出所有山的参数，求天际线长度。地平线不算入天际线，天际线由山的上轮廓组成。山的数量 ≤ 100.

6.8.4 关键词

计算几何

6.8.5 算法讨论

考虑将山劈成两半计算，那么就得到了一堆线段。因为线段数不超过 200，所以考虑枚举线段，然后计算它没有被覆盖的长度。在这里比较适合的算法是扫描法。使用扫描法时有以下几种情况：情况一要算出交点的位置，判断一下

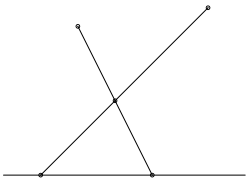

Figure 13: 情况一

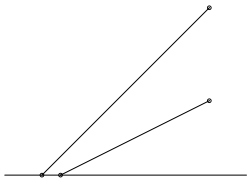

Figure 14: 情况二

左边还是右边被覆盖；情况二只需判断是否被覆盖。分开两种情况，只要算出两条线段在共同区间的端点的高度即可：情况一为两端点高低情况相反，否则为情况二。扫描时只需统计横轴的长度，统计完后用勾股定理就可以算出线段的长度。

6.8.6 时空复杂度

- 时间：$O(n^2)$

- 空间：$O(n)$

6.9 2008K: Steam Roller

6.9.1 题目编号

2008 K

6.9.2 题目名称

Steam Roller

6.9.3 题目大意

给出一个网格图，每条边有一个权值，若权值为零则不可通过。

现在要从起点到达终点。在每一次向一个方向的连续移动上，除了头和尾两条边，其他边经过时间为边的权值。剩下的边经过时间为权值的两倍。

求最段时间。

$1 \leq n, m \leq 100$。

6.9.4 关键词

最短路

6.9.5 算法讨论

把一个点拆成 8 个点，分别表示最后一条边在哪 4 个方向上，头尾和中间的状态。

也就是说，0 到 3 号点最后一条边的费用为两倍，4 到 7 号点最后一条边的费用为边权。

然后，转移一方有中间状态时，方向不能变化。转移的费用按照题意容易得出。

另外，边不用随着拆点而增加，只要用 $v(i, j, k)$ 表示点 (i, j) 向方向 k 的边的权值即可。

采用 dijkstra 之后复杂度就是 $O(n^2 \log n)$ 了。

6.9.6 时空复杂度

- 时间：$O(n^2 \log n)$
- 空间：$O(n^2)$

7 ACM/ICPC World Finals 2007

7.1 2007A: Consanguine Calculations

7.1.1 题目编号

2007 A

7.1.2 题目名称

Consanguine Calculations

7.1.3 题目大意

考虑控制血型的两对等位基因，分别是 ABO 和 Rh。ABO 的性状有四种：

基因	性状
AA	A
AO	A
BB	B
BO	B
AB	AB
OO	O

Rh 的性状有两种：

基因	性状
++	+
+−	+
−−	−

给出父母、孩子其中两人的性状，求另一人可能的性状。

7.1.4 关键词

模拟、枚举

7.1.5 算法讨论

将两种基因分开讨论。

- 枚举父母有什么基因，判断是否符合知道的性状。若另一人为父母，则需要判孩子是否可能出现；否则直接判可能出生的孩子的性状可行。
- Rh 基因因为只有两个，所以简单 If 一下就可以判断出另一人可能表现的 Rh 基因性状。

做完两个性状之后，要组合起来输出。

7.1.6 时空复杂度

- 时间：$O(1)$
- 空间：$O(1)$

7.2 2007C: Grand Prix

7.2.1 题目编号

2007 C

7.2.2 题目名称

Grand Prix

7.2.3 题目大意

给出一个由一些首尾相连的有向线段组成的图案和参数 θ，并规定对于点 p，$h(p) \propto x_p \cdot \sin\theta$。做以下操作：

- 判断图案是否符合：对于每条有向线段，均有 h(终点) $\geq h$(起点)。
- 若不符合，判断是否能通过旋转图案使之满足要求。
- 若能通过旋转满足要求，则输出旋转的最小角度和旋转方向。

给出 n 个点以描述这个图案。第一条有向线段为 $(0,0) \to (x_1, y_1)$，第 i 条有向线段为 $(x_{i-1}, y_{i-1}) \to (x_i, y_i)$ $(i \geq 2)$。

$n \leq 10^4$。

7.2.4 关键词

计算几何

7.2.5 算法讨论

这道题目看起来就像纯粹的几何题。

- 因为 θ 是常数，所以，当 $\theta > 0^\circ$ 时，$h(p) \propto x_p$，也就是让点的 x 坐标单调不减；当 $\theta = 0^\circ$ 时，图案无论如何都会符合要求。
- 考虑旋转之后 y 轴正方向的单位向量 $\vec{m}$。那么若满足要求，则有每条有向线段的方向向量 $\vec{e_i} \times \vec{m} \geq 0$。
- 那么，就可以考虑通过 n 条有向线段的方向向量求出 $\vec{m}$ 的角度区间。
  - 首先，第一个方向向量会限制出一个 π 大小的区间。

  - 然后，其他方向向量就分为两类；第一类在 π 大小的区间内，另一类不在。
  - 两类分别取最严的限制向量：第一类，逆时针方向上的更严；第二类相反。
  - 得出两个向量之后，需要判断无解：第一类的向量叉积第二类的结果非负。
  - 然后就可以对得出的两个最严的限制向量使用反三角函数，得出角度区间。
- 最后，加上各种 If 判断即可。

7.2.6 时空复杂度

- 时间：$O(n)$
- 空间：$O(n)$

7.3 2007E: Collecting Luggage

7.3.1 题目编号

2007 E

7.3.2 题目名称

Collecting Luggage

7.3.3 题目大意

要从平面上的一个点出发到达终点。移动的最大速率为 V。终点在一个简单多边形边界上向逆时针方向移动，移动的速率恒定，且小于 V。移动到终点的途中，不能进入简单多边形内部。求最短时间。

多边形点数 $n \leq 100$。

7.3.4 关键词

二分、最短路、计算几何

7.3.5 算法讨论

因为终点在移动，所以考虑将终点“停”下来。

二分到达终点的时间——因为终点移动速率小于人的移动速率，所以二分性成立。那么剩下的就是最短路。

明显，最短路是由一条条直线线段组成的，两端是顶点或起点。但是并不是任意两点之间都能走直线。

考虑枚举直线线段，并判断合法性。那么就有两种情况：

- 跨过边：这种情况下，明显线段不合法。
- 经过点：预处理每个顶点出发，朝向外面的角度范围。那么，判断时分两种情况：
  - 顶点为线段端点：判断从这个顶点出发的线段的角度是否在范围内。
  - 顶点非线段端点：与上一情况类似，但是需要判断两个角度 α 和 $\alpha + \pi$。

可以先用 dijkstra 处理出起点到每个顶点的最短路。然后二分答案，找到终点的位置，再用 $O(n^2)$ 时间找到能直接到达它的顶点或起点，并算出最短距离。

7.3.6 时空复杂度

- 时间：$O(n^3)$[23]
- 空间：$O(n^2)$

[23]主要复杂度在建图。

7.4 2007G: Network

7.4.1 题目编号

2007 G

7.4.2 题目名称

Network

7.4.3 题目大意

要求将一些信息输出。每条信息分成了一些信息包，每个包有三个特征值：属于哪条信息，包含了信息中的哪一段。

对于一条信息，它的信息包的输出必须连续，而且按照包中含有的信息的顺序输出。

现在，有一些包被顺序读入，为了满足上面的要求，有些信息包会被存入缓冲区。当然，也可以不存入缓冲区就输出。

问在最好的信息输出顺序下，缓冲区最小的大小。

信息个数 $n \leq 5$，信息包个数 $m \leq 1000$。

7.4.4 关键词

暴力、贪心、离散化

7.4.5 算法讨论

因为 n 很小，所以考虑暴力枚举信息的输出顺序。

- 枚举信息输出顺序后，要用贪心计算缓冲区最小的大小。
- 用两个指针，一个指向读入的信息包，一个指向应该输出的下一个信息包。
- 随着第一个指针的移动，第二个指针尽量移动，贪心地让当前在缓冲区的信息包最少。
- 移动时维护在缓冲区的信息包的总大小即可。

另外，为了让第二个指针顺利移动，需要知道每个信息包是所属信息第几个输出的信息包。这一步用离散化即可。

7.4.6 时空复杂度

- 时间：$O(n! \cdot m)$
- 空间：$O(nm)$

7.5 2007I: Water Tanks

7.5.1 题目编号

2007 I

7.5.2 题目名称

Water Tanks

7.5.3 题目大意

有一排水箱，相邻的水箱之间有水管连接。除了一号水箱，其它水箱都不直接与大气连通。水箱的底面积都是 1 平方米，水管的直径可以无视，同时水管能同时通过水和空气，从一号水箱开始，水管的高度递增。给定水箱各自的高度，水管各自的高度，往一号水箱灌水，求这些水箱最多能容纳多少立方米的水。$2 \leq$ 水箱数 ≤ 10.

水和空气遵循如下的物理规律：

1. 水往下流。
2. 与大气连通的区域气压恒等于一个大气压。
3. 空气可以压缩，水不能压缩。
4. 一个封闭空间中，气压处处相等。在任何情况下，等质量的气体，体积与气压的乘积为定值。
5. 水深为 D 时的水压等于：水表面气压加上 $0.097 \cdot D$ 个大气压。
6. 在一个静止的连通水体中，同一高度下水压处处相同。

7.5.4 关键词

模拟

7.5.5 算法讨论

先不考虑一号水箱，其他水箱流水的过程最多有三个阶段：

1. 水深还没超过前边的水管高度；上方气体与前一个水箱、后面所有水箱的气体连通。
2. 水深超过了前边的水管但是没超过后面的水管；上方气体与后面所有水箱的气体连通。
3. 水深超过了后面的水管；上方气体拥有独立的封闭空间。

程序需要对这三个阶段进行模拟，同时要判断一个水箱能否到达下一个阶段。

阶段一 判断是否能到达下一阶段（利用下一阶段的水压判断），可以则到下一阶段，否则结束并判断这个水箱会有多少水。

阶段二 判断水是否能流到下一水箱（同上）；可以则开始处理下一水箱的阶段一，然后判断是否能到阶段三；否则结束并计算水量。

阶段三 计算水量并结束。

计算水量的时候需要解二次方程。

7.5.6 时空复杂度

- 时间：$O(n)$
- 空间：$O(n)$

7.6 2007J: Tunnels

7.6.1 题目编号

2007 J

7.6.2 题目名称

Tunnels

7.6.3 题目大意

有一个无向图，某人从点 1 出发要到点 0，你能在任何时候瞬间删掉上面没有人的某些边。目标是让某人到不了点 0. 求最少的删边数。点数 ≤ 51，边数 ≤ 1000.

7.6.4 关键词

最小割、图论

7.6.5 算法讨论

这题看上去像最小割，但是考虑样例第二个数据，可以发现，不在一开始就删完最小割效果可能更优。设 $f(i)$ 表示，假如某人从 i 点出发，最少要删掉多少条边。那么有如下算法。

1. $f(i)$ 的初始值设为无穷大。
2. 枚举 i，如果存在一个 y，$f(j) > y$ 的点组成的图中点 i 到点 0 的最小割为 x，$x + y$ 可以更新 $f(i)$ 则更新。
3. 若第二步中没有更新则结束，否则继续第二步。

因为删掉 x 条最小割的边之后，肯定要经过 $f(j) \leq y$ 的点，这样就可以再删 $f(j)$ 条边使得某人到不了点 0.

这个算法的优化就是，$f(j) > y$ 的点组成的图，随着 y 的减少，原来的流信息仍可用。所以每次第二步实际只用做 n 次最小割。

7.6.6 时空复杂度

- 时间：$O(k \cdot nm)$
- 空间：$O(n + m)$

8 ACM/ICPC World Finals 2006

8.1 2006A: Low Cost Air Travel

8.1.1 题目编号

2006 A

8.1.2 题目名称

Low Cost Air Travel

8.1.3 题目大意

每次旅行要求使用飞机航线从某个城市开始，按顺序经过一些城市。

每条航线经过若干个城市；每次使用航线，要从航线的起点出发，但是可以在航线经过的任一个城市离开，同时以后都不能继续这一次飞航。

同一条航线可以在一次旅行中多次使用，费用重复计算。

航线数 $m \leq 20$，每条航线或每次旅行经过城市数 $k \leq 10$。

8.1.4 关键词

最短路

8.1.5 算法讨论

考虑用 dijkstra 等算法实现最短路。

- 状态表示：用 $f(i, j, k)$ 表示当前在第 i 条航线第 j 个点，顺序经过了旅行的前 k 个城市。
- 那么转移就有两种：
  - 继续使用当前航线；那么就走到当前航线的第 $j + 1$ 个点。
  - 离开当前航线转机；枚举一条航线，若在这条航线的起点，就可以转到这条航线。
- 转移时，顺便维护一下 k 值，就可以采用最短路算法计算。

8.1.6 时空复杂度

- 时间：$O(k^2 m \log km)$
- 空间：$O(k^2 m)$

8.2 2006B: Remember the A La Mode!

8.2.1 题目编号

2006 B

8.2.2 题目名称

Remember the A La Mode!

8.2.3 题目大意

有相同总数的大饼和冰淇淋，大饼有 n 种，冰淇淋有 m 种，每种可以有多个。

现在要将大饼和冰淇淋搭配出售（你懂的，甜筒）。不同种类的配合利润不一样；并且有些种类甚至不能搭配。

求最小利润和最大利润。

$1 \leq n, m \leq 50$。

8.2.4 关键词

费用流、二分图

8.2.5 算法讨论

这题是明显的二分图带权完备匹配。因为同种冰淇淋或大饼同种的有不止一个，所以为了简化，考虑比较一般的模型。

- 那么就可以考虑费用流。
- 建图很简单：把大饼和冰淇淋分开，一边连源，一边连汇，容量为数量，费用为零；可以搭配的种类间连边，容量无穷，费用为收益。
- 那么算法就是[24]：
  1. 构图；
  2. 用 spfa 跑一遍最小费用最大流；
  3. 重构图；
  4. 改变比较函数，再用 spfa 跑一遍最大费用最大流。

[24]或许 KM 算法 · 改能解决这种问题，但是某并不会。

8.2.6 时空复杂度

- 时间：$O(n^4)$
- 空间：$O(n^2)$

8.3 2006D: BipartiteNumbers

8.3.1 题目编号

2006 D

8.3.2 题目名称

BipartiteNumbers

8.3.3 题目大意

给出一个数 x，求另一个数，满足：

- 大于 x，且为 x 的倍数；
- 为一个“二段数”，十进制位上有且仅有两种数字，各自连成一段。例如 8844 是二段数，8848 不是，8888 也不是。

$1 \leq x \leq 99999$。

8.3.4 关键词

暴力、二分

8.3.5 算法讨论

因为数学分析比较难以入手，所以考虑优化暴力。

- 首先，末尾为零的数，答案的两种数字其中一种必是零。于是就可以优化：
  - 枚举另一种数字长度、和数字本身。
  - 那么，因为在末尾添加零不会影响倍数关系，所以零的最少个数可以二分。
  - 因为只考虑模 x 的余数，所以可以预处理全为 1 的数的余数，用于计算；并且每种数字的长度不超过 x。
  - 另外，得到临时答案之后可以用最优性剪枝加快速度。
- 对于末尾不为零的数，只能按顺序枚举“二段数”判断余数。利用上面提到的“预处理全为 1 的数”的方法，可以大大降低常数。

尽量优化之后，就可以通过了。

8.3.6 时空复杂度

- 时间：$O(?)$
- 空间：$O(n)$

8.4 2006E: Bit Compressor

8.4.1 题目编号

2006 E

8.4.2 题目名称

Bit Compressor

8.4.3 题目大意

一种 01 串的压缩方法是，选择一段连续的 1，两端外是 0 或者空，然后把这段 1 替换为这段长度的二进制值。

压缩时，若压缩后不会变短，就不会压缩；并且压缩过的地方不再压缩。

现在给出压缩后的 01 串，原串的长度和 1 的个数，问属于下面那种状态：

- 不存在符合的原串能压缩为目标串。
- 存在唯一一个符合的原串。
- 存在多个符合要求的原串。

8.4.4 关键词

动态规划

8.4.5 算法讨论

因为状态表示比较明显，所以考虑动态规划。

- 考虑设状态 $f(i,j,k)$，表示当前还原出的串长度为 i、0 的个数为 j、给出串的前 k 位均被还原，是否能到达。
- 那么，转移就使用记忆化搜索，直接从 $f(i,j,k-1)$ 转移或者枚举一端为第 k 位的“二进制串”还原。
- 如此，空间就是 $O(Ln^2)$，时间就是 $O(Ln^3) > 10^9$。
- 但是，考虑每一个状态对应着给出串一些区间的选取；那么最坏情况下有约 13 个区间可以选取，同时区间可以合并，这样，就可以估算为 3^{13} 个状态。这个规模是可以接受的；另外，因为区间不能太大，并且用三进制代表后不能出现“20”这种组合，所以实际规模远小于 3^{13}。因此算法的实际效率非常好。

- 因为状态数其实很少，所以可以用哈希或者 map 存储状态。

8.4.6 时空复杂度

- 时间：$\ll O(Ln^3)$
- 空间：$\ll O(Ln^2)$

8.5 2006G: Pilgrimage

8.5.1 题目编号

2006 G

8.5.2 题目名称

Pilgrimage

8.5.3 题目大意

杰克要管理帐目。他带着一群人，有时会总共支出一些钱，有时向所有人收同样多的钱；另外有些时候人会离开或加入。

对于人员变动，杰克总会把钱平分，离开的就分钱，加入的就要交钱。

如果杰克的帐目恰好没有涉及分数，那么给出帐目的一部分，求这一部分开始时可能有多少人。

帐目只记录 4 种操作：PAY、COLLECT、IN、OUT；每个操作后面会有一个数。PAY 表示这次总共支出多少钱，COLLECT 表示这次每人收了多少钱，IN 表示这次有多少人进来，OUT 表示这次有多少人离开。

数据组数 ≤ 30000，帐目长度 ≤ 50，一次 PAY 的数目 ≤ 2000，所有在账本上的数目都是正整数。

8.5.4 关键词

枚举

8.5.5 算法讨论

考虑什么时候会对人数有限制。

在一次 IN ／ OUT 之后，接着一次 PAY，然后再一次 IN ／ OUT 操作；这样，假设一开始有 x 个人，第一次 IN ／ OUT 操作的数目是 p，PAY 的数目是 q，那么有 $(x+p)|q$。

同时观察可以发现，COLLECT 操作是没有任何影响的，两个 IN ／ OUT 操作之间的 PAY 是可以合并的。那么如此就得到了一条条类似 $(x+a)|b$ 的式子。

于是就可以枚举出一个式子的解，代入到其他式子进行验证。这样就可以算出答案。

8.5.6 时空复杂度

- 时间：$O(T \cdot k^{0.5} \cdot n)$
- 空间：$O(n)$

8.6 2006I: Degrees of Separation

8.6.1 题目编号

2006 I

8.6.2 题目名称

Degrees of Separation

8.6.3 题目大意

给出一副图，每个点有一个没有空格的名字，判断图是否连通，连通则求图的直径。

8.6.4 关键词

图的遍历

8.6.5 算法讨论

首先需要处理点的名字。可以用 Hash 或 map 套 string。这样就能建立名字和点的序号的对应关系，从而转化为方便处理的形式。

接着判图的连通，从一个点开始遍历一下，若能遍历整副图则连通。然后考虑枚举直径的一端，BFS 求出离这个点最远的距离，即可。

8.6.6 时空复杂度

- 时间：$O(n^2 + nm)$
- 空间：$O(n + m)$

8.7 2006J: Routing

8.7.1 题目编号

2006 J

8.7.2 题目名称

Routing

8.7.3 题目大意

给出一个有向图，要求保留最少的点，使得点 1 与点 2 能互相到达。

$n \leq 100$。

8.7.4 关键词

最短路

8.7.5 算法讨论

因为要让一个包含点 1、点 2 的环点数最少，所以考虑将环分成两部分。那么就变成了求两条路径。

为了方便，让两条路径都从 1 出发。假设第一条路径为 1 到 2，考虑设 $d(i,j)$ 表示两条路径的最后一个点分别为 i 和 j，求最短路。如果两路径没有交，那么很简单，但是如果有交，就有下图两种情况：

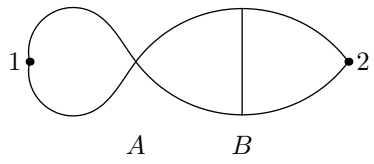

Figure 15: 相交的情况

情况 A： 对于这种情况，当从 $d(i,j)$ 转移到 $d(i,i)$ 或 $d(j,j)$ 时，移动的费用减一即可。

情况 B： 这种情况比较棘手。增加一种转移，从 $d(i,j)$ 转移到 $d(j,i)$，费用为 i 到 j 的距离减一。因为假如最后点互换的话，两条路径在此处的边也相交。预处理增加 floyd 即可。

求最短路用 dijkstra 时，复杂度就是 $O(n^3 \log n)$。

8.7.6 时空复杂度

- 时间：$O(n^3 \log n)$
- 空间：$O(n^2)$

9 ACM/ICPC World Finals 2005

9.1 2005B: Simplified GSM Network

9.1.1 题目编号

2005 B

9.1.2 题目名称

Simplified GSM Network

9.1.3 题目大意

对于常见的手机，它需要地面基站的支持。

一般，手机会与离当前位置最近的基站通讯；所以在使用者移动时，手机可能会切换通讯的基站。

给出 B 个基站的位置，C 个城市的位置，R 条直线道路连接的两个城市。Q 次询问，每次问从一个城市沿着 R 条道路到另一个城市最少需要切换多少次基站。

$1 \leq B, C \leq 50$，$0 \leq R \leq 2500$，$Q \leq 10$。

9.1.4 关键词

计算几何、最短路

9.1.5 算法讨论

假若能够处理出每条道路会切换多少次，那么就是一个最短路问题。

- 考虑基站之间的关系；可以发现，由这些基站可以得出一个 voronoi 图；穿过一次 voronoi 图的边，就切换了一次基站。
- 当然，我们可以直接用半平面交、分治等求出 voronoi 图；但是，这样做的编程复杂度太高了，得不偿失。
- 因为 voronoi 图中的边必是两个基站中垂线的一部分，所以考虑对于每条道路，枚举所有中垂线。
- 算出中垂线与道路的交点之后，就判断交点是否在 voronoi 图的边上：由 voronoi 图的定义，若交点在边上，离交点最近的两个点就是中垂线的两个点。

- 这样，就可以算出每条道路的费用，然后用 floyd 等最短路算法即可。

9.1.6 时空复杂度

- 时间：$O(R \cdot C^3)$
- 空间：$O(C^2 + R)$

9.2 2005C: The Traveling Judges Problem

9.2.1 题目编号

2005 C

9.2.2 题目名称

The Traveling Judges Problem

9.2.3 题目大意

有一幅地图，一些人要从一些城市到一个城市。移动需要租车，租车会产生费用。他们租车所遵循的规则如下：

1. 租一辆车的费用与它行驶的距离成正比。
2. 所有车的费用与行驶距离的比例相同。
3. 一辆车可以容纳任意数量的乘客。
4. 任意一对城市之间最多只有一条道路直接相连，每条道路都是双向的且长度大于 0。
5. 每个人的起始城市到目标城市都至少有一种路线。
6. 若多个人的路线中经过同一城市，则这些人从该城市到目标城市必乘同一辆车。
7. 一个人可以乘一辆车到某个城市，再乘另一辆车离开该城市。

求最少的费用，和同时访问城市总数最少、同时访问集合的字典序最少的任意方案。城市数 $n \leq 20$，出发的人数 $k \leq 10$。

9.2.4 关键词

最短路、状压

9.2.5 算法讨论

可以发现，最优解的路线显然没有环，是一棵树，所以第六条其实是废的。那么用 $f(s,i)$ 表示当前车子上的人的集合是 s，在第 i 个城市所耗费的费用。转移只有两种：

车子移动 这种转移下 s 并不会变化，是单纯的最短路，采用 Floyd 或 bellman-ford 都可以。

换车 这种情况下，只会出现在一个城市的所有人跑到同一部车上。所以枚举 s 的一个子集 t，用 $f(t,i)$ 和 $f(t \oplus s,i)$ 更新 $f(s,i)$ 即可。

算出 $f(s,i)$ 之后，要搜索一下方案树，让树的大小和字典序最小，然后再输出。

9.2.6 时空复杂度

- 时间：$O(2^k \cdot n^3)$
- 空间：$O(n \cdot 2^k)$

9.3 2005E: Lots of Sunlight

9.3.1 题目编号

2005 E

9.3.2 题目名称

Lots of Sunlight

9.3.3 题目大意

有一列公寓楼排成东西方向的一列。每栋楼每层只有一个公寓。把阳光直射到整个东边的墙壁、西边的墙壁或在正上方，视为太阳直射了这个公寓。给出所有公寓楼的位置、层数，求一些公寓一天内被直射的时间。太阳被视为等角速度移动，所有公寓的高度、宽度是一样的。公寓楼数 ≤ 100，询问数 ≤ 1000.

9.3.4 关键词

枚举

9.3.5 算法讨论

对于两栋公寓楼，太阳高度角小于一定角度的时候，一栋楼的阴影会把另一栋的一些或全部公寓遮住。比如下图中，太阳在虚线延长线左方的时候，第二栋公寓楼最上面那个公寓没有被直射（实际上第二栋公寓楼所有公寓都没有被直射）。

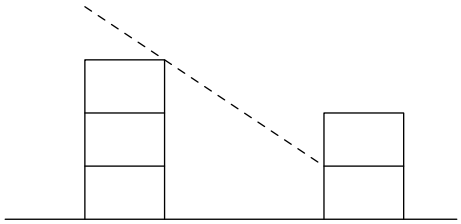

Figure 16: 示例

因此，考虑分别枚举东方和西方的楼，算出直射时太阳高度角的上下限，最后转换为时间输出即可。

9.3.6 时空复杂度

- 时间：$O(Q \cdot n^2)$
- 空间：$O(n)$

9.4 2005G: Tiling the Plane

9.4.1 题目编号

2005 G

9.4.2 题目名称

Tiling the Plane

9.4.3 题目大意

给出一些简单多边形，只能复制和平移，求是否能铺满平面。给出的简单多边形的边平行于 X 轴或 Y 轴。边数 ≤ 50，多边形周长 ≤ 50。

若一个上述的简单多边形能以上述方法铺满平面，那么必满足以下两个条件之一：

i 存在四个点 A、B、C、D，按顺序分布在周长上；周长上 AB 段经平移可与 CD 段重合，BC 段与 DA 段同理。

ii 存在六个点 A、B、C、D、E、F，按顺序分布在周长上；周长上 AB 段经平移可与 DE 段重合，BC 与 EF、CD 与 FA 重合。

9.4.4 关键词

搜索

9.4.5 算法讨论

观察上面两个条件，共同点就是，连续的一半数量的段组成了周长的一半。那么考虑枚举一个开始点，然后搜索。

以图17作为示意图。令圆表示整个周长，PN、QM 均为直径，P 为某一段的始点，那么 NM 就是 PQ 对应的一段。为了使得 PQ 平移之后与 NM 重合，$P \to Q$ 一段在几何上与 $M \to N$ 一段相等，在多边形的表示上仅边的方向相反。

使用上述方法检验两段是否重合，就能枚举每一段长度地搜索，得出答案。

9.4.6 时空复杂度

- 时间：$O(n^4)$

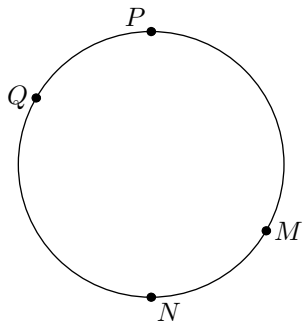

Figure 17: 示意

- 空间：$O(n)$

9.5 2005H: The Great Wall Game

9.5.1 题目编号

2005 H

9.5.2 题目名称

The Great Wall Game

9.5.3 题目大意

在一个 $n \times n$ 的棋盘上，有 n 颗棋子。

棋子可以向上下左右移动，求让所有棋子排成一条直线的最少步数。

另外，沿对角线排列也算直线。

$n \leq 15$。

9.5.4 关键词

二分图带权匹配

9.5.5 算法讨论

因为 n 不大，所以考虑枚举所有可能的直线，然后就能转化为二分图上的问题。

- 枚举了直线之后，问题就是每一颗棋子移动到那一个位置。

  Proof. 考虑棋子向位置连边，费用为曼哈顿距离。那么每一个完备匹配都能对应一个相同费用的实际方案：若一个棋子沿曼哈顿距离最小的路径移动中不可避免地撞上了其他棋子；这时两个棋子就可以交换最终位置继续移动，费用不变。 □

- 那么，构图之后，就是一个二分图最小权匹配。
- 这样，就可以使用KM 算法，$O(n^3)$ 地解决一次完备匹配；因为枚举直线是 $O(n)$ 的，所以最终复杂度为 $O(n^4)$。

9.5.6 时空复杂度

- 时间：$O(n^4)$
- 空间：$O(n^2)$

9.6 2005I: Workshops

9.6.1 题目编号

2005 I

9.6.2 题目名称

Workshops

9.6.3 题目大意

要安排一些会议到一些房间里。会议有人数和持续时间，房间有人数上限和能被使用的时间。求最小的没有被安排到房间里的会议数，和在这种情况下最少的不在房间里的人数。房间数、会议数 ≤ 1000.

9.6.4 关键词

贪心

9.6.5 算法讨论

将房间按时间排序，然后从短到长枚举房间，每个房间贪心地取能取的人数最多的会议。

贪心的证明. 分情况讨论。

- 在贪心中，房间的时间是递增的，当前房间选的会议的持续时间是没有关系的，所以考虑人数。人数小的会议在后面能安排的房间更多，所以取最多人的房间比少人的房间优，选取的会议数不会变少。
- 假如最优解人数更多，那么就有会议 p 只出现在最优解。考虑最优解容纳 p 的房间，贪心解中必能取到不劣于它的会议；那么这就要求最优解中有其他独有会议更优，但是这样又返回了前面，所以最优解不能更优。

□

9.6.6 时空复杂度

- 时间：$O(n^2)$
- 空间：$O(n)$

9.7 2005J: Zones

9.7.1 题目编号

2005 J

9.7.2 题目名称

Zones

9.7.3 题目大意

有 n 个区，区之间可能有公共部分。每部分都会有一些人。现在要从这 n 个区域里取 m 个，使得包含的人数最大，并且集合的字典序最小。$n \leq 20, m \leq 10$.

9.7.4 关键词

枚举

9.7.5 算法讨论

这题咋一看像是容斥，但实际上不是。因为题目已经给出了每个公共区域内的人数，所以只要算出枚举集合内覆盖这个公共区域的区域数，把多的区域减掉。算交集的大小可以用 $O(2^n)$ 预处理数组，这样复杂度就从 $O(nm \cdot 2^n)$ 变成 $O(m \cdot 2^n)$ 了。但是因为要判断答案的字典序，所以复杂度还是 $O(n \cdot 2^n)$.

9.7.6 时空复杂度

- 时间：$O(n \cdot 2^n)$
- 空间：$O(2^n)$

10 ACM/ICPC World Finals 2004

10.1 2004A: Carl the Ant

10.1.1 题目编号

2004 A

10.1.2 题目名称

Carl the Ant

10.1.3 题目大意

蚂蚁卡尔经常毫无理由的蜿蜒着走，有时候会无数次穿越自己的路。当其他蚂蚁到达一个交叉点的时候，它们总是沿着气味最强或者说最新的路走。

蚂蚁长 1 厘米，每秒移动或打洞 1 厘米，准确沿着它们的路径走（在走过角落的时候转 90 度）。如果两只蚂蚁在同一个瞬间到达同一点，在卡尔的路上走过了较长距离的蚂蚁优先移动；如果两只蚂蚁不是在同一个瞬间到达同一点，等待时间最长的蚂蚁优先移动。

卡尔先挖洞爬到地面上，在原点于时刻 0 出发。然后它沿着它的路径走到终点，再挖洞回到地底。其他的蚂蚁以一定的时间间隔跟着卡尔。给出卡尔的路径的描述以及其他蚂蚁的出发时间，你需要计算所有蚂蚁走完给定路径所需的时间。保证所有蚂蚁都可以走完。

数据组数 ≤ 20，线段数 ≤ 50，蚂蚁数 ≤ 100，蚂蚁出现间隔 ≤ 100，坐标绝对值 ≤ 100.

10.1.4 关键词

模拟

10.1.5 算法讨论

对于每只蚂蚁，我们要维护头／尾的位置、方向、移动距离、等待时间。对于卡尔走路留下的信息，记录在地图矩阵上比较方便。还没有到出发时刻的蚂蚁不放到处理的列表里方便一些。

处理每只蚂蚁的移动的时候，因为蚂蚁是同时移动的，所以当一只蚂蚁停下来的时候，可能会造成一定规模的塞车。因此为了避免蚂蚁的重叠，最好采用类似冒泡的处理方式，循环直到没有蚂蚁停下来为止。

注意蚂蚁离开的时间是从地上看不到的时间，也就是说蚂蚁刚钻进洞口时（尾部还在地面）还没有离开。

10.1.6 时空复杂度

- 时间：$O(m^2 \cdot L)$
- 空间：$O(m^2)$

10.2 2004E: Intersecting Dates

10.2.1 题目编号

2004 E

10.2.2 题目名称

Intersecting Dates

10.2.3 题目大意

题目描述了一种服务，能够给出任意日期区间内的股票信息。但是服务的费用与查询区间长度成正比。

已知之前询问了 m 次，现在要获得 n 个区间内的股票信息。

明显，直接使用服务很可能会让成本增加。所以，你需要根据已经查询过的区间，得出这次要向服务询问的区间。

$0 \leq n, m \leq 100$。

10.2.4 关键词

扫描法

10.2.5 算法讨论

因为这是一个在一维（时间）上的关于区间的问题，所以考虑扫描法。

- 对于一个区间，将其拆成两个点，一个为正，一个为负；就能维护当前点在多少区间内了。
- 若在任一个旧询问内，则无需向服务询问，所以要维护当前在多少旧询问区间内。
- 当然，只有新询问内的才有询问意义；那么就也要维护新询问。
- 根据当前点在新旧询问区间的状态，就可以判断这个点是否需要询问。
- 因为直接将日期映射为自然数比较麻烦，另外“日期表示”的数字可以直接比较日期的先后，所以考虑用“表示数字”直接排序离散化。

另外，比较麻烦的一点是，离散化之后，为了输出，要计算一个日期的前一个日期或后一个日期。

10.2.6 时空复杂度

- 时间：$O(n \log n)$
- 空间：$O(n)$

10.3 2004G: Navigation

10.3.1 题目编号

2004 G

10.3.2 题目名称

Navigation

10.3.3 题目大意

给出一些信号源，这些信号源以一定的速度往各自的方向移动。每个信号源在某个瞬间会发出信号，然后接收器在同一瞬间接收到了这些信号（信号的速度固定）。因为可能误差，所以规定将距离在 0.1 内的两个点视为同一个点。给出信号源的位置信息和发送时间，判断接收器对于目的地的状态：

1. 信号异常，无可能的接收器位置。
2. 有多个可能的接收器位置。
3. 还没到目的地，输出目的地方向。
4. 到达目的地。

信号源个数 ≤ 10，信号源的速率均为 100，信号的速率为 350。

10.3.4 关键词

数学[25]

10.3.5 算法讨论

这道题目的关键是接收器的位置。那么一个简单的思路就出来了：

1. 随便找两个相交的圆求交点。
2. 验证交点是否为接收器位置。

验证只要扫一遍求距离即可。如此，就要求两个圆的交点。当然我们可以解二元二次方程，但是有另外一些比较直观的方法。

[25] ACM 数学题难度差异挺大…

将圆心连线，两交点连线。这两线段明显垂直，所以设两圆心到垂足的距离分别为 a、b，半弦长为 c，就有：

$$a^2 + c^2 = r_1^2$$
$$b^2 + c^2 = r_2^2$$
$$\Downarrow$$
$$(a+b)^2 - r_1^2 + r_2^2 = 2b \cdot (a+b) \tag{2}$$

事实上有两种情况，如图18和19。

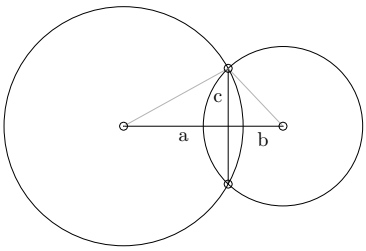

Figure 18: 情况一

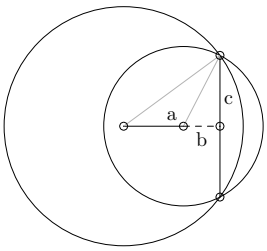

Figure 19: 情况二

这样，在情况一中，a、b 都为非负，而在情况二中，a、b 中有一个为负[26]。如此，求出 a、b 中一个和 c 之后，就可以用向量加减求出两个交点。

注意最后要考虑 0.1 的误差[27]。

10.3.6 时空复杂度

- 时间：$O(n^2)$
- 空间：$O(n)$

[26]其实这两种情况如何对求交点没太大影响，只是说明不能直接开方求 a、b。

[27]数据似乎很厚道——因为 0.1 误差其实可以做出很多 trick，但是这道题只要最后判一下就可以了。

10.4 2004H: Tree-Lined Streets

10.4.1 题目编号

2004 H

10.4.2 题目名称

Tree-Lined Streets

10.4.3 题目大意

题目要求在一些街道上种树。这些街道都是一些二维平面上的线段，且并不会在端点处相交。树之间至少间隔 50 米，离十字路口至少 25 米。求最多可以种多少树。街道数 ≤ 100，$0 \leq$ 坐标 ≤ 100000.

10.4.4 关键词

计算几何

10.4.5 算法讨论

首先，这道题的一个 trick 就是：树不需要离街道端点 25 米远（当然样例就可以看出来了）。因为树要远离十字路口，所以线段之间的交点把线段上的树分成一段一段的。

假如得到了一段路，中间没有十字路口，那么在这段路上可以种的树的棵树是可以直接算出来的——减去不能种的长度除以 50 再加 1. 这样，这道题的重点就在于求交点。判断线段相交用叉积即可，黑书上有。

求交点：设线段为 AB, CD，那么设交点为 $A + x\overrightarrow{AB}$，就有

$$x\overrightarrow{AB} = \overrightarrow{AC} + y\overrightarrow{CD}$$

将二维向量视为复数，令 $\overrightarrow{AB} = a, \overrightarrow{CD} = b, \overrightarrow{AC} = c$，那么变形得

$$x = \frac{\overline{a} \cdot c + y \cdot \overline{a} \cdot b}{a \cdot \overline{a}}$$

因为 $x, a \cdot \overline{a}$ 都是实数，所以上式右边分子虚部为零。令 $\times$ 表示叉积运算，那么有：

$$\mathbf{a} \times \mathbf{c} + y \cdot (\mathbf{a} \times \mathbf{b}) = 0$$

$$y = -\frac{\mathbf{a} \times \mathbf{c}}{\mathbf{a} \times \mathbf{b}}$$

这样求出 y，就可以轻易求出交点。求出交点之后排一下序，就可以扫一遍得出每段路的长度。

10.4.6 时空复杂度

- 时间：$O(n^2 \cdot \log n)$
- 空间：$O(n)$

10.5 2004I: Suspense!

10.5.1 题目编号

2004 I

10.5.2 题目名称

Suspense!

10.5.3 题目大意

有两座楼，要在其间架一座桥。桥面平行于地面，上方有两条平行的抛物线形状的缆索，与桥面有无穷多的绳子连接，吊起了桥面。缆索的最低点离桥面恰好 1 米；缆索两端连接着两个指定的窗口的最低端；桥面至少距离两个指定窗户最低点两米，地面 1 米。同时，两座楼的规格是一样的，每层高 3 米，窗口最低端里地板 1 米。

每层楼有一些人家养了猫或鸟。猫能往上跳小于 0.5 米，往下跳小于 3 米。为了安全，不能让猫能遇见鸟。注意，猫不会考虑回家的事情。求缆索最长的长度。数据组数 $\leq 10^5$，$2 \leq$ 楼层数 ≤ 25，$1 \leq$ 楼间距 ≤ 25。

10.5.4 关键词

数学

10.5.5 算法讨论

因为 3 米是一层的高度，所以猫能往下跳一层；因为要缆索的长度最大化，所以桥面会尽量低。综上，桥面的高度只可能是某一层窗户最低端高度、加 0.5 米、减 0.5 米。于是，枚举答案判断就可以得出桥面的高度。

算出桥面的高度之后，需要计算抛物线。为了方便，考虑设抛物线为 $y = k \cdot x^2$。那么有

$$\frac{k \cdot x^2}{k \cdot (L-x)^2} = \frac{h_1}{h_2}$$

就可以算出抛物线、抛物线最低点位置。

为了计算抛物线长度，考虑微分长度 l，又由勾股定理，得

$$\mathrm{d}l = \sqrt{\mathrm{d}y^2 + \mathrm{d}x^2} = \sqrt{\frac{\mathrm{d}y^2}{\mathrm{d}x^2} + 1} \cdot \mathrm{d}x$$

那么抛物线长度就是

$$\int_{x_0}^{x_1} \sqrt{4kx^2+1} \cdot \mathrm{d}x$$

要计算这个积分，可以去查一下积分表，有下面的公式：

$$\int \sqrt{a^2+x^2} \cdot \mathrm{d}x = \frac{1}{2}x\sqrt{a^2+x^2} + \frac{1}{2}a^2 \ln|x+\sqrt{a^2+x^2}| + C$$

运用这个公式和上面的结论即可算出抛物线的长度。

10.5.6 时空复杂度

- 时间：$O(T \cdot n)$
- 空间：$O(n)$

11 ACM/ICPC World Finals 2003

11.1 2003A: Building Bridges

11.1.1 题目编号

2003 A

11.1.2 题目名称

Building Bridges

11.1.3 题目大意

给出一幅地图。地图由很多方格组成，每个格子有一个信息表示这里是建筑或不是。一个建筑物是八连通的。现在要用一些直线把建筑物尽量直接或间接连起来。直线只能沿着方格的边缘伸展；若两条直线相交或部分重合，视为互不干扰。最多的建筑物连起来的同时，直线数要最小，然后要求直线总长度最小。地图大小 $\leq 50 \times 50$。

11.1.4 关键词

最小生成树

11.1.5 算法讨论

首先，考虑将地图信息抽象成图。

1. 先染色一遍，把所有建筑物找出来，得到格子的所属。
2. 然后枚举是建筑物的格子；每个这样的格子向左、向上分别射出两条线，碰到其他有建筑的格子就停下，连边，退出；明显，直线不会继续延伸，因为从碰到的格子出发肯定更优。

然后，这就是一个显然的最小生成树问题。考虑时间复杂度。因为每个格子最多向上、向左分别连 3 个格子，所以边数是 $O(n^2)$ 级别的。同时，点数也是 $O(n^2)$ 级别的；因此，用 prim 是会超时的，要用 kruskal 才可能过。另外要注意常数优化。

11.1.6 时空复杂度

- 时间：$O(n^2 \log n)$

- 空间：$O(n^2)$

11.2 2003B: Light Bulbs

11.2.1 题目编号

2003 B

11.2.2 题目名称

Light Bulbs

11.2.3 题目大意

给出一串 n 个灯泡，每个灯泡上都有一个开关。每个开关影响与正下方灯泡距离不超过 1 的灯泡。也就是说改变两端的开关，会有两个灯泡的状态改变，改变中间的开关会使三个灯泡改变。给出初始状态和末状态，求一个改变方案，改变开关次数最少，改变的序列字典序最小。输入输出都用十进制数表示二进制数，二进制数每个数位表示一个灯泡的状态。输入的数位数 ≤ 100.

11.2.4 关键词

高精度、递推

11.2.5 算法讨论

首先，十进制和二进制的转化需要高精度。然后问题就是怎么求出答案二进制数。观察可以发现，确定了 1 到 i 的开关状态，第 i 个灯泡的状态只跟第 $i+1$ 个开关有关。于是我们可以用异或和运算求出第 $i+1$ 个开关的状态。如此，我们就可以枚举第一个开关的状态，然后递推出其他 $n-1$ 个开关的状态。

11.2.6 时空复杂度

时间复杂度主要在于高精度。

- 时间：$O(n^2)$
- 空间：$O(n)$

11.3 2003G: A Linking Loader

11.3.1 题目编号

2003 G

11.3.2 题目名称

A Linking Loader

11.3.3 题目大意

给出一些链接的描述，求十六位校验和与符号表。

一个对象模块有序地包含 0 个或多个外部符号定义、0 个或多个外部符号引用、0 个或多个字节的代码和数据（可能包含对外部符号值的引用）以及一个模块结尾标志。在这个问题中，一个对象模块表示为一个文本行序列，每行以一个大写字母开头，描述了剩余文本的含义。

I. D symbol offset : “D”语句是一个外部符号定义。它定义了 symbol 的地址为当前模块的代码和数据的第一个字节地址向后偏移 offset 字节。symbol 是一个长度小于等于 8 的大写字母字符串。offset 是一个至多 4 位的十六进制数（使用大写字母 A-F）。在一组测试数据中，这样的定义至多为 100 个。

II. E symbol : “E”语句是一个外部符号引用。它表明 symbol 的值可能会在当前模块的代码或数据中被调用（可能被定义在另一个对象模块中，并且可能会在当前模块之后）。每一个模块中，“E”语句从 0 开始连续编号，使它们能够在“C”语句中被引用。

III. C $nbyte_1byte_2 \ldots byte_n$: “C”语句指定了当前模块的代码和数据的第一个或接下来 n 个字节的地址。数值 n 是一个一位或两位的十六进制数，并且不超过十六进制的 10。每一个 byte 是一个一位或两位的十六进制数，或者是一个“$”符号。一个“$”符号后方跟随一个字节（中间不存在换行），表示引用当前模块中一个从 0 开始编号的外部符号。编译器将这个符号的值（即它的地址）插入当前链接程序所指的位置（即“$”符号所代表的地址以及后一个地址），十六进制数的高位放在“$”所指示的位置。其余的字节（不紧接在任一个“$”后的）将会被放入连续的内存地址中，起始位置为第一个未使用的内存地址。如果这个被引用的符号从未被定义，那么将其地址视为 $0000_{(16)}$。

IV. Z : 一行一个字母“Z”代表了当前模块的结束。

十六位校验和的计算方法如下：首先将其值设为 0，然后以升序遍历所有地址，每次循环左移一位，然后加上该内存地址所存储的值，并无视溢出。循环左移的定义为，将十六位校验和的二进制的最高位移至最低位，其他位左移一位。比如 FF00 循环左移一位之后得到 FE01。

保证不存在超过四位十六进制数的地址，每行的格式都按照以上所述并且不存在语法错误。

11.3.4 关键词

模拟

11.3.5 算法讨论

这是一道略麻烦的模拟题。要点如下：

i. 首先就是十六进制和十进制的问题，这个手写可以，用 format 也可以。
ii. 因为“D”语句的定义可能在引用之后，所以要把对所有 symbol 的引用放入一个表，最后扫一遍。
iii. “E”语句必会在对应的“C”语句引用前出现，所以可以放心按读入顺序做。
iv. 因为“E”语句中用到的 symbol 不一定会有，所以最后要确保都在载入表内。
v. “C”语句中，每个 symbol 会占用两个字节； “D”中定义的 symbol 只是一个地址，并不占存储空间。
vi. 事实上，每个组数据所占用的内存都是一个连续的块，而校验和的计算只要从这个块的开头扫到结尾。

11.3.6 时空复杂度

- 时间：$O(n \log n)$
- 空间：$O(n)$

11.4 2003H: A Spy in the Metro

11.4.1 题目编号

2003 H

11.4.2 题目名称

A Spy in the Metro

11.4.3 题目大意

某人在一些站点间移动。这些站点排成一排，从左、从右均有发车往另一端。每两个相邻站之间的移动时间给定，某人要从最左边出发，到某个特定时候出现在最右边的站点，求他在站点上等待的最短时间。他可以往回乘车。站之间的移动手段只有坐车。站点数 ≤ 50，每一端出发的车辆数 ≤ 50，数据组数 ≤ 1500.

11.4.4 关键词

动态规划

11.4.5 算法讨论

考虑用 $f(i,j)$ 表示状态；i 表示当前在 i 站点，j 表示这个时刻 j 号车到达了站点 i，状态表示这种情况下等待时间总计多少。状态的转移有两种，一种是继续搭这辆车，第二种是等待其他车到这个站台再搭上去。

另外，要预处理所有状态的时间顺序。值得一提的是，第二种转移暴力 $O(n)$ 是可以过的，但是也可以预处理出同时到达的、接下来到达的第一部车子，进而让 DP 时间复杂度变为 $O(n^2)$。

11.4.6 时空复杂度

因为要对状态的时间排序，所以一组数据需要 $O(n^2 \log n)$ 的时间。

- 时间：$O(T \cdot n^2 \log n)$
- 空间：$O(n^2)$

11.5 2003I: The Solar System

11.5.1 题目编号

2003 I

11.5.2 题目名称

The Solar System

11.5.3 题目大意

给出两个行星运动轨迹的半长轴和半短轴，还有其中一个的运行周期。要求运用开普勒三定律求得另一个行星在近日点移动若干时间后的位置。开普勒三定律见 维基。输入在 4 字节有符号整数范围内，除了时间为非负整数，其他都是正整数，精度要求尽量高。

11.5.4 关键词

二分、数学

11.5.5 算法讨论

利用开普勒第三定律，可以算出另一行星的周期；再利用开普勒第二定律，可以知道这个行星运行了要求的时间之后，与焦点连线扫过的面积。由椭圆的标准方程

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

可以用以下式子表示在椭圆上的一个点：

$$x = a \cdot \cos\theta$$
$$y = b \cdot \cos\theta$$

那么，为了计算答案就需要二分 θ，计算扫过的面积。考虑如何计算扫过的面积。先考虑没有超过椭圆面积一半。这时有两种情况。
可以见得，两种情况的扫过的面积都与类半弓形 AMN、$\triangle PNM$ 有关。以情况一为例：扫过面积 $=$ 类半弓形$AMN + \triangle PNM$。然后情况二中，式子的加减号相反。但实际上两条式子可以合成一个，两种情况其实是相当的。

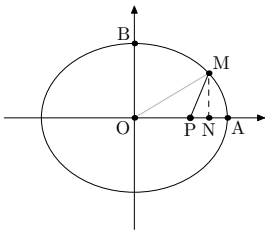

Figure 20: 情况一

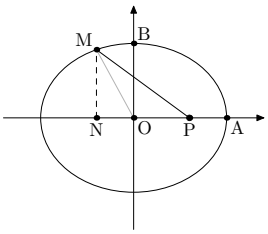

Figure 21: 情况二

如此，就需要计算类半弓形 AMN 的面积。我们利用积分观察：

$$\begin{aligned} S_{\text{半弓形}AMN} &= \int_{X_N}^{X_A} \frac{b \cdot \sqrt{a^2 - x^2}}{a} \cdot dx \\ &= \frac{b}{a} \cdot \int_{X_N}^{X_A} \sqrt{a^2 - x^2} \end{aligned}$$

可以看出积分式代表一个半径为 a 的圆的一个半弓形。那么就可以计算了。

扫过的面积超过一半的时候，利用补集转化即可。另外，注意精度[28]。

11.5.6 时空复杂度

- 时间：$O(\log C)$
- 空间：$O(1)$

[28]某人真的很坑爹…

11.6 2003J: Toll

11.6.1 题目编号

2003 J

11.6.2 题目名称

Toll

11.6.3 题目大意

在一个地方，商人运送货物到达任何一个地方必须抽税；离开则不会。

抽税的规则是这样的：

- 进入任何一个村庄，只拿走一件货物。
- 进入任何一个城镇，取走货物数除二十取上整数目的货物。

对于村庄，用小写字母‘a’到‘z’表示；对于城镇，用大写字母‘A’到‘Z’表示。在城镇或村庄间，有若干条道路。

有一个商人，从一个地方到另一个地方，需要运送 p 件货物到目的地；求出发时最少的货物数。

11.6.4 关键词

最短路

11.6.5 算法讨论

从正面考虑，发现难以入手；因此考虑反过来，从终点退回起点。

- 设 $f(i)$ 表示 i 到终点至少需要多少货物。
- 那么转移的费用就是进入 i 所抽的税。这个用枚举、解方程什么的计算都可以。
- 然后，对于这个模型就可以使用 dijkstra 等算法计算 f。

11.6.6 时空复杂度

- 时间：$O(n \log n)$
- 空间：$O(n^2)$

12 ACM/ICPC World Finals 2002

12.1 2002A: Ballons in a Box

12.1.1 题目编号

2002 A

12.1.2 题目名称

Ballons in a Box

12.1.3 题目大意

给出一些点和一个盒子，你要做一些操作。每次操作选取一个在盒子内，不在任一球体内的给出的点，膨胀出一个尽量大的球体，与其他球体没有重合部分，没有超出盒子的部分。求经过一系列操作后盒子内最小的剩余空间。给出点数 ≤ 6.

12.1.4 关键词

枚举

12.1.5 算法讨论

由于操作产生的球体半径与之前产生的球体有关，需要考虑对点操作的顺序。因为最多只有 6 个点，所以暴力枚举所有 $6! = 720$ 个顺序就可以了。枚举之后注意判断是否能膨胀球体的条件；另外球体的体积公式是 $V = \frac{4}{3}\pi r^3$.

12.1.6 时空复杂度

- 时间：$O(n \cdot n!)$
- 空间：$O(n)$

12.2 2002E: Island Hopping

12.2.1 题目编号

2002 E

12.2.2 题目名称

Island Hopping

12.2.3 题目大意

在一个平面上，有一些点。现在要用线段连接一些点，使得所有点连通，并且线段总长最小。

因为最优方案可能有不止一个，所以任意方案均能被接受。

现在，同时画每一条线段，1 时间单位画 1 长度。每个点均有一个权值，问与 1 号点连通的时间的加权平均数。

点数 $n \leq 50$。

12.2.4 关键词

最小生成树、贪心

12.2.5 算法讨论

一看，似乎需要枚举所有最优方案；但是，事实上所有最优方案的加权平均数都是一样的。

Proof. 因为关键是“连通”，所以考虑 kruskal 算法的过程。kruskal 算法用并查集记录连通情况；因为多种方案只可能出现在边权相同的时候，所以可以发现，无论相同边权时选取哪些边，选完这个边权之后的连通情况是固定的。同时，对于每个点，有关计算的是加入 1 的并查集时候所在的边权。于是结论显然正确。 □

现在需要计算连通第 i 个点与 1 号点最小的路径上最长的边。于是，考虑采用类似 prim 的算法。

- 设 $d(i)$ 表示当前连通块连接 i，最小的路径最长边长度。
- 一开始，当前连通块为 1 号点。

- 找到 d 最小的一个点 k，加入当前连通块。
- 然后 1 号点就可以通过 k 连通其他点，就可以更新 d 值；然后继续找点。
- 另外，因为要通过 k 连通 1 号点，所以更新的值不小于 $d(k)$。

算法正确性可由 kruskal 算法过程得证。

12.2.6 时空复杂度

- 时间：$O(n^2)$
- 空间：$O(n)$

12.3 2002G: Partitions

12.3.1 题目编号

2002 G

12.3.2 题目名称

Partitions

12.3.3 题目大意

对矩形的划分定义一种偏序关系。一个矩形的划分是指把一个矩形分成若干个较小的、不重叠的子矩形。

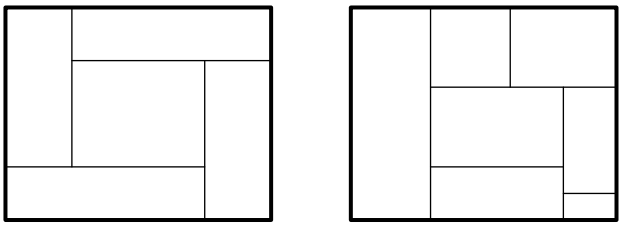

Figure 22: 一些例子

对于同一矩形的两个划分，若划分 A 可以通过 B 继续划分得来，那么说 A 比 B 精细，即 $A < B$。

那么对于偏序关系，就有求上确界和求下确界。下图左边两个划分的上确界为第三个划分，下确界为第四个划分。

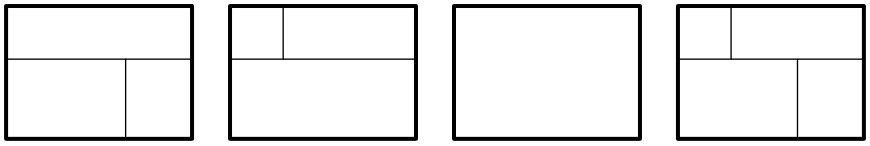

Figure 23: 上下确界的例子

给出两个同一矩形划分，子矩形的边长为整数，求上确界和下确界。

划分的矩形边长 ≤ 20。

12.3.4 关键词

数学

12.3.5 算法讨论

其实这是两个问题。

- 下确界：下确界很好求，只要将两个划分的线段的存在性“或”一下，就可以得到。证明显然。
- 上确界：首先，类似下确界的思路，将线段的存在性“且”一下，就会得到一个图形。但是这个图形不一定是划分。

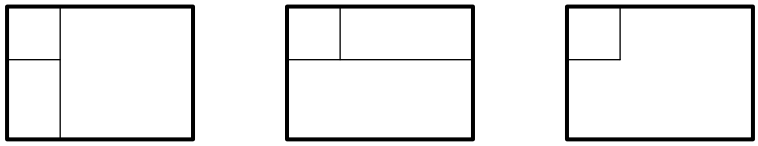

Figure 24: 一个反例

不过，显然上确界可以通过这个图形去掉一些线段得到。

因为都是整点，所以可以将图形转化为一个网格图。不合法时，必存在至少一个格点，它与相邻的边形成一个角（如图24），或者它的度数为一(删掉图24的角之后)。

那么就有一个算法：找到这些不合法的格点，删掉与之相连的所有边。这个算法的实现可以用 DFS 或者类似冒泡的方式。冒泡的实现稍微容易一些，但是复杂度较高[29]。

12.3.6 时空复杂度

- 时间：$O(n^4)$[30]
- 空间：$O(n^2)$

[29] n 只有 20，怎样也没关系。
[30] 若用 DFS 的实现方式可做到 $O(n^2)$。

12.4 2002H: Silly Sort

12.4.1 题目编号

2002 H

12.4.2 题目名称

Silly Sort

12.4.3 题目大意

给出一列数，每次可以交换任意两个数的位置，费用为两个数的和。求从小到大排好序所需的最小费用。所有数两两不同，个数 $n \leq 1000$，数的大小 ≤ 1000。

12.4.4 关键词

贪心

12.4.5 算法讨论

因为没有相同的数，所以最后每个数的位置是确定的。这样原数列和目标数列就构成了一个置换。关于置换的含义可以看维基。

考虑置换中的每一个轮换。若要这个轮换的费用最小，有两种方案：

- 找到轮换中最小的元素，反方向走一遍这个轮换，使得轮换完成；费用为最小值 $\times$ (轮换大小 -2) + 轮换所有元素之和。
- 找到数列中最小的元素，利用它参与到轮换的交换过程中：先与轮换中最小的交换，然后逆轮换方向走一遍，最后交换回原来的位置。这样费用为轮换最小值 + 轮换所有元素之和 + 最小值 $\times$ (轮换大小 $+1$)。

对每个轮换贪心的取这两个方案的最小值即可。

12.4.6 时空复杂度

- 时间：$O(n \log n)$
- 空间：$O(n)$

13 ACM/ICPC World Finals 2001

13.1 2001B: Say Cheese

13.1.1 题目编号

2001 B

13.1.2 题目名称

Say Cheese

13.1.3 题目大意

在一个三维空间里，有一些球体。在这些球体里移动不需要时间，球体外每移动一毫米耗时 10 秒。求一个点到另一个点的最短时间。球体数 ≤ 100, 球体半径 $\leq 10^4$, 坐标绝对值 $\leq 10^4$.

13.1.4 关键词

最短路

13.1.5 算法讨论

因为在球体里移动不需要时间，所以考虑将球体看成一个点。这样，点之间的距离就是欧几里德距离减去两个点的半径与零取 MAX——将起点终点视为半径为零的点。如此，三维空间就抽象成了一个完全图。求最短路用 Floyd 算法即可。

13.1.6 时空复杂度

- 时间：$O(n^3)$
- 空间：$O(n^2)$

13.2 2001F: A Major Problem

13.2.1 题目编号

2001 F

13.2.2 题目名称

A Major Problem

13.2.3 题目大意

给出两个大调记号，问一些音符从某大调到另外一个大调中对应什么音符。以 C 大调和 D♭ 大调为例。

C 大调	C	D	E	F	G	A	B	C
D♭ 大调	D♭	E♭	F	G♭	A♭	B♭	C	D♭

由此，C 大调中的 F 对应 D♭ 中的 G♭。输入的大调记号可能不合法，例如 B♯。

大调满足以下规律：

1. 一个大调前七个音符的字母互不相同，最后一个和第一个音符一样。
2. 一个大调为“全音 -全音 -半音 -全音 -全音 -全音 -半音”；全音：在循环的十二平均律里距离为 2，半音：在循环的十二平均律中距离为 1.
3. 一个大调中不会同时出现 ♭ 和 ♯。

十二平均律如下：

序号	1	2	3	4	5	6	7	8	9
音符	C/B♯	C♯/D♭	D	D♯/E♭	E/F♭	F/E♯	F♯/G♭	G	G♯/A♭
序号	10	11	12						
音符	A	A♯/B♭	B/C♭						

Table 1: 十二平均律

13.2.4 关键词

音乐、模拟

13.2.5 算法讨论

这道题考的真的是音乐。

如果学过大调，能背出来，那么任务很简单，就是打常量表然后模拟一下[31]。

如果没学过，那么就要去找规律。用各种方法可以知道[32]，音符的选取是有顺序的，比如 C 大调是 1-3-5-7-8-10-12-1[33]. 又因为每个大调是唯一的，所以大调的音律在 Table 1中是从左往右循环的。于是就可以预处理出所有的大调，然后模拟。

13.2.6 时空复杂度

Q 是询问的规模。

- 时间：$O(Q)$
- 空间：$O(1)$

[31] 当然你可以去维基。

[32] 5 个全音，2 个半音，恰好跨过了 12 个音律，因此可能有最多两种方案，字母是循环、按顺序变化的。可惜某打了搜索才发现。

[33] 在 Table 1中。

14 ACM/ICPC World Finals 2000

14.1 2000A: Abbott's Revenge

14.1.1 题目编号

2000 A

14.1.2 题目名称

Abbott's Revenge

14.1.3 题目大意

这是一个游戏。有一个至多 9 的网格，要求从起点往初始方向移动后到达终点的一个最短的方案。每个点规定了，从某一个方向到达这个点之后，是否能直走、左转、右转。数据组数 ≤ 1000.

14.1.4 关键词

搜索

14.1.5 算法讨论

明显可以设状态为 $f(i,j,k)$，表示现在在 (i,j)，朝向为 k。因为状态很少，所以广搜、记忆化都可以。这题特别的一点就是有 Trick：

- 起点终点可能一样，但是一出生要先往初始方向移动[34]。
- 最优解中，可能再次回到出生时的状态[35]。

14.1.6 时空复杂度

- 时间：$O(T \cdot 9^2 \cdot 4)$
- 空间：$O(9^2\dot{4})$

[34]只有 tsinsen 上有这个 Trick…
[35]同上…

14.2 2000C: Cutting Chains

14.2.1 题目编号

2000 C

14.2.2 题目名称

Cutting Chains

14.2.3 题目大意

有一些环，其中有一部分相连。现在要让这些环连成任意的一条链。求最少让多少环打开。

14.2.4 关键词

暴力

14.2.5 算法讨论

明显，一个环在打开后，就可以拿出来，连接至少两个部分链，并在最后的链成型前不会闭合。于是考虑暴力枚举哪些环需要打开。

确定哪些环打开之后，需要判断是否可行，条件如下：

1. 除去了打开的环，其他连通块必须为一个点或一条链。原因明显。
2. 剩下的连通块个数减一不大于打开的环数。因为需要用打开的环连接剩下的部分。

知道判断条件之后，就可以最优 $O(n)$ 的做一次判断了。

14.2.6 时空复杂度

- 时间：$O(2^n n)$
- 空间：$O(2^n)$

14.3 2000E: Internet Bandwidth

14.3.1 题目编号

2000 E

14.3.2 题目名称

Internet Bandwidth

14.3.3 题目大意

给出一个网络，每条边双向且两个方向流速相等。求起点到终点的最大流速。点数$n \leq 100$，边数$m \leq \frac{n(n-1)}{2}$，每条边流速 ≤ 1000.

14.3.4 关键词

网络流

14.3.5 算法讨论

这道题目是最大流的经典模型。并不需要将每条无向边拆成两条有向边，只要在初始的残量网络里，每条边两个方向的流量赋值为一样的即可。最大流用 dinic、SAP 什么的应该都可以过。

14.3.6 时空复杂度

- 时间：$O(n^2 \cdot m)$
- 空间：$O(n + m)$

14.4 2000F: Page Hopping

14.4.1 题目编号

2000 F

14.4.2 题目名称

Page Hopping

14.4.3 题目大意

给出一幅图，求，最短路径的平均长度；图保证连通，边权均为一。虽然结点编号在区间 $[1, 100]$，但是结点编号不一定是从 1 到 n。$1 \leq$ 结点数 ≤ 100，边数 $\leq 10^4$。

14.4.4 关键词

最短路

14.4.5 算法讨论

因为图比较特殊，边权均为一，所以求最短路的时候可以用 BFS。当然，如果想用 dijkstra,floyd,bellman-ford,spfa 之类的也没关系。

求出所有最短路长度之后，加起来，除以 $n \cdot (n-1)$ 即可得出平均数。

14.4.6 时空复杂度

- 时间：$O(n^3)$
- 空间：$O(n^2)$

15 ACM/ICPC World Finals 1999

15.1 1999A: Bee Breeding

15.1.1 题目编号

1999 A

15.1.2 题目名称

Bee Breeding

15.1.3 题目大意

给出一个无限大的蜂巢，以顺时针顺序编号，如图25。每次询问给出两个编号，每一步移动只能移向边相邻的蜂巢 [36] ，问这两个蜂巢的最短距离是多少。询问次数 $T \leq 100$，编号 ≤ 10000.

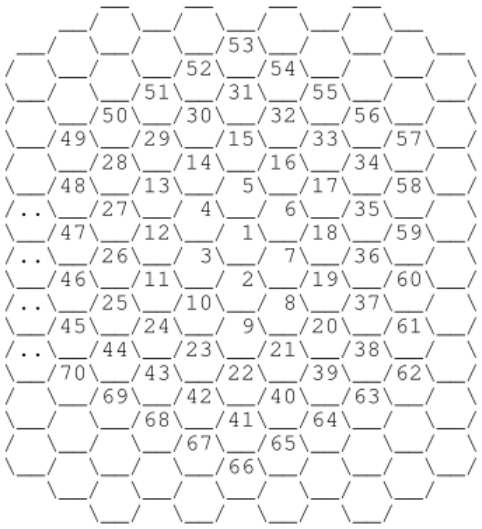

Figure 25: 蜂巢

[36]例如巢室 19 和 30 之间的距离为 5。一条连接两个巢室的最短路径为 19 - 7 - 6 - 5 - 15 - 30。

15.1.4 关键词

数学[37]、模拟

15.1.5 算法讨论

首先，蜂巢的六边形结构有点难搞。因此，考虑转化到坐标系上。将 $1 \to 7$ 作为 x 轴正方向；将 $1 \to 3$ 作为 y 轴正方向；把 1 视为原点。考虑每一层蜂巢，它们就会在坐标系上成为这个样子，如图26。

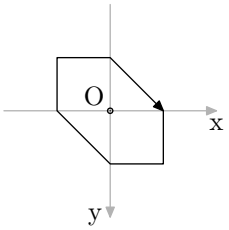

Figure 26: 一层蜂巢

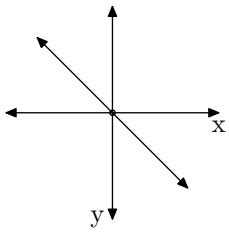

Figure 27: 移动的方向

同时，向相邻蜂巢移动也使得坐标系上的移动不一般，如图27。这样，算法就是：处理编号，将一层蜂巢分成六段，得出坐标；然后判断一下两个点的相对位置，得出移动的最短距离。

15.1.6 时间复杂度

- 时间：$O(T)$
- 空间：$O(1)$

[37]找规律？

15.2 1999C: A Dicey Problem

15.2.1 题目编号

1999 C

15.2.2 题目名称

A Dicey Problem

15.2.3 题目大意

有一个骰子，放在一个地图上，遵循着一定规律滚动；问一条滚出起点再滚回来的路径。骰子的展开图如下：

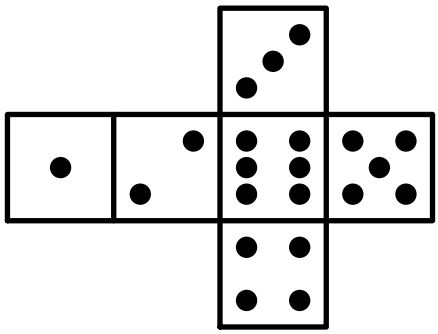

Figure 28: 展开图

滚动的规则：地图上每个格子都有一个数字，骰子只能滚到数字与骰子顶部数字相同的格子或数字为负一的格子。数据保证不会有多解，给出骰子的初始状态和地图，求路径。地图的长宽 ≤ 10。

15.2.4 关键词

搜索

15.2.5 算法讨论

骰子的状态可以用上面和正面的数字来表示，但是滚动的时候需要知道侧面的数字是什么。那么就需要预处理出上面和正面是什么数字时侧面的数字是什么。预处理有很多方法，其中全手算略有点麻烦。

搜索的时候，因为保证了最多一组解，所以只要 DFS；DFS 到解就直接弹出 DFS、输出答案。DFS 的时候维护一下骰子的状态、外部栈里存一下骰子的位置。这道题目主要麻烦是预处理，搜索过程并不难写。

15.2.6 时空复杂度

- 时间：$O(6^2 \cdot n^2)$
- 空间：$O(6^2 \cdot n^2)$

15.3 1999E: Trade on Verweggistan

15.3.1 题目编号

1999 E

15.3.2 题目名称

Trade on Verweggistan

15.3.3 题目大意

一些商人要进货，每次购入货堆上的最顶上的一些货物。每个货物的入价都不尽相同，但是商人对于每个货物都卖 10f[38]. 对于每一次进货，有若干个货堆，给出所有货物的价格，求商人的最大获利和要买货物的数量（若有超过 10 个解则输出最小的 10 个解）。进货次数 ≤ 10，每次进货货堆数 ≤ 50，每个货堆的货物数 ≤ 20.

15.3.4 关键词

贪心、动态规划

15.3.5 算法讨论

因为只能取最上面的若干个，所以可以预处理出取若干个之后的利润。然后贪心得到这堆货的最大利润，和可取货物数量。如此，考虑用 $f(i,j)$ 表示，前 i 个货堆得到最大利润，共买了 j 个货物，是否可行。这样，就可以像背包一样转移，就能得到所有可能的买货数量。

15.3.6 时空复杂度

w 表示货堆数，b 表示每堆的箱子数。

- 时间：$O(w^2 \cdot b^2)$
- 空间：$O(w \cdot b)$

[38]文艺复兴时期佛罗伦萨所用货币弗罗林。

15.4 1999G: The Letter Carrier's Rounds

15.4.1 题目编号

1999 G

15.4.2 题目名称

The Letter Carrier's Rounds

15.4.3 题目大意

题目要求你描述一些基于 SMTP 的发生在 MTA 之间的通信。一个 SMTP 会话开始之后，发件 MTA 一条一条地发送命令给接收 MTA，在接收 MTA 返回一个三位的返回码之后发送下一条命令。*命令及返回码如下：*

HELO myname 发件 MTA 名字

MAIL FROM:<sender> 发件者地址

RCPT TO:<user> 一位收件者的地址

DATA 正文的传送开始，传送结束用只有一个. 的一行标记。

QUIT 结束会话

221 关闭会话

250 无异常

354 开始传送邮件

550 无此用户

如果一个接收 MTA 有多个收件者，RCPT TO 命令就会有多个。一个邮件发送给在不同 MTA 的收件者需要各自独立的 SMTP 会话。输入 MTA 名字与其中的用户名、每份邮件的收发地址及正文。输出每个 SMTP 会话，会话内容缩进。SMTP 会话的顺序与邮件收发地址中 MTA 出现顺序相同。保证收发地址中出现的 MTA 名字都是有效的。MTA 名字与用户名长度不超过 15，邮件正文每行不超过 75 个字符，有效地址数、邮件正文总行数、邮件数均不超过 10^5.

15.4.4 关键词

模拟、字符串处理

15.4.5 算法讨论

这题是带字符处理的模拟题，要点如下：

- 读入所有名字之后，处理出所有类似 user@MTA 这样的地址，用哈希压起来，用以判断收件地址的有效性。
- 对于每封邮件，读入之后先把 MTA 取出来并去重，接着按出现顺序处理会话。
- 发送完收件地址并判断有效性之后，判断一下是否输出正文。
- 正文的处理就是每个换行后加 5 个空格。因为这题是全文比较，所以不能有多余的空格。

15.4.6 时空复杂度

- 时间：$O(n + output)$
- 空间：$O(n)$

16 ACM/ICPC World Finals 1998

16.1 1998D: Page Selection by Keyword Matching

16.1.1 题目编号

1998 D

16.1.2 题目名称

Page Selection by Keyword Matching

16.1.3 题目大意

每个页面拥有一些关键字。每次搜索，都会按顺序给出一些关键字用于搜索。一个页面和一次搜索间，有关系强度。假如页面第 i 个关键字与搜索第 j 个关键字相同，那么强度增加 $(N-i+1)\times(N-j+1)$。N 是一个页面或一次搜索所允许关键字最大数目。

现在，动态的新增页面，并进行搜索，问每次关系强度前五的页面是什么。页面数 $n\leq 25$，关键字长度 ≤ 20，$N=8$。关键字忽略大小写。

16.1.4 关键词

暴力

16.1.5 算法讨论

因为 $n\leq 25$，所以考虑对于每次搜索，暴力枚举页面并算出强度。

考虑对关键字进行 Hash，并且每个页面或搜索进行排序。然后，就能够每个页面 $O(N)$ 的扫一次算出强度。算出所有强度之后，排序即可输出。

注意关键字大小写忽略。

16.1.6 时空复杂度

- 时间：$O(QnN)$
- 空间：$O(nN)$

16.2 1998E: Petri Net Simulation

16.2.1 题目编号

1998 E

16.2.2 题目名称

Petri Net Simulation

16.2.3 题目大意

给出一个有向图，图中有两种点。库所存储令牌，变迁点让输入库所减少令牌，输出库所增加令牌。

每次操作，输入库所到变迁点的每条边都移动一个令牌，要保证移动到变迁点之后合法；然后变迁点到输出库所的每条边移动一个令牌。也就是说，可以把变迁点看作有无穷令牌的特殊点。

每次操作只会对一个变迁点进行，有多个可行则任意。求操作给定次数之后的状态，若中途“死”掉则输出“死亡时间”[39]。库所数 $n \leq 100$，变迁点数 $p \leq 100$，边数 $m \leq 20000$，要求操作次数 $Q \leq 1000$。

16.2.4 关键词

模拟

16.2.5 算法讨论

直接按照题意进行模拟，注意理解每次操作时，每条有关的边都要移动一个令牌。还有就是，操作了指定次数之后，不需要判断可不可以进行下一次，因为操作完后都视为活着。时间最大约为 $m \cdot Q = 2 \times 10^7$，是足够的。

16.2.6 时空复杂度

- 时间：$O(m \cdot Q)$
- 空间：$O(m)$

[39] 若干次操作之后可能无法进行下一次操作。