第4章 数学计算

吴永辉
ACM-ICPC Asia Programming Contest 1st Training Committee – Chair
Shanghai Normal University，ACM-ICPC Asia Training League
yhwu@fudan.edu.cn
WeChat : 13817360465

目录

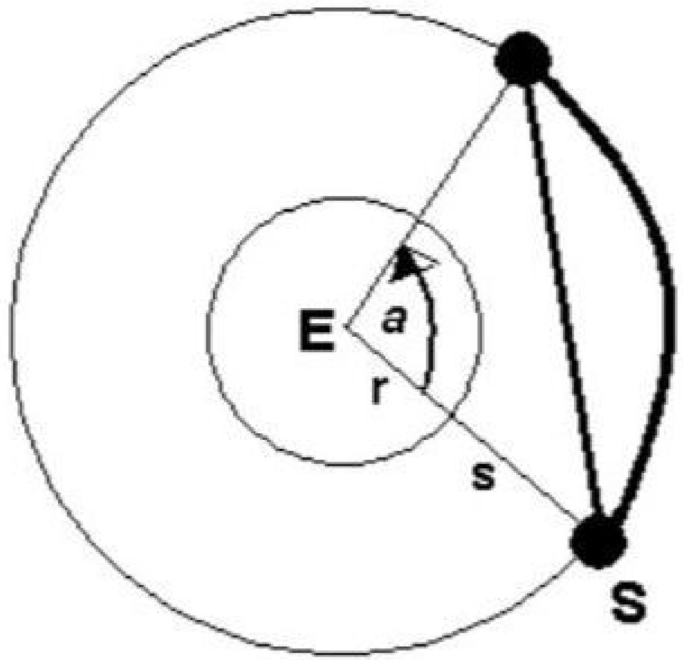
4.1几何初步

- 本节给出运用平面几何、立体几何和解析几何的知识，编程解决问题的实验。

4.1.1 Satellites

- 试题来源： **THE ROCKFORD PROGRAMMING CONTEST 2001**
- 在线测试：**UVA 10221**

地球的半径是 6440 公里。有许多人造卫星围绕着地球运行。如果两颗人造卫星对地球中心形成一个夹角，您能计算出这两颗人造卫星之间的距离吗？距离分别以圆弧距离（arc distance）和直线弦距离（chord distance）来表示。这两颗人造卫星是在同一轨道上（本题设定这两颗人造卫星是在一条圆形路径上，而不是在椭圆路径上，绕地球运行）。



E = Earth　S = Satellite

- 输入
- 输入包含一个或多个测试用例。
- 每个测试用例一行，给出两个整数*s*和*a*，以及一个字符串”min”或”deg”；其中*s*是人造卫星与地球表面的距离，*a*是这两颗人造卫星对地球中心的夹角。以分(′)，或者以度(◦)，为单位。输入不会既给出分，又给出度。
- 输出
- 对于每个测试用例，输出一行，给出两个卫星之间的圆弧距离和直线弦距离，以公里为单位。距离是一个浮点数，保存小数点后的六位数字。

试题解析

角度的单位是度和分，绕圆一周为 360 度，1 度可以再细分成 60 分。已知圆心角的角度 *angle*（以度为单位），$0 \le angle \le 180$ 度，和半径 r，可以计算圆弧距离 *arc_dist* 和直线弦距离 *chord_dist*：$arc\_dist = 2\pi \times r \times \frac{angle}{360}$；$chord\_dist = r \times \sin\left(\frac{angle \times \pi}{2 \times 180}\right) \times 2$。如果 *angle* 以分为单位，则 $arc\_dist = 2\pi \times r \times \frac{angle}{360 \times 60}$，

$$chord\_dist = r \times \sin\left(\frac{angle \times \pi}{2 \times 180 \times 60}\right) \times 2$$。

对于本题，要注意给出的夹角大于 180 度的情况。

由于本题求解过程使用三角函数 sin，在 C++中使用 sin，就要加上#include<cmath>。

4.1.2 Fourth Point !!

- 试题来源： **The World Final Warmup (Oriental) Contest 2002**
- 在线测试： **UVA 10242**

- 给出平行四边形中两条相邻边的端点的(x, y)坐标，请找到第四个点的(x, y)坐标。

- 输入
- 输入的每行给出8个浮点数：首先，给出第一条边的一个端点和另一个端点的(x, y)坐标；然后，给出第二条边的一个端点和另一个端点的(x, y)坐标；所有的坐标均以米为单位，精确到毫米。所有的坐标的值在-10000和+10000之间。输入以EOF终止。
- 输出
- 对于每行输入，输出平行四边形的第四个点的(x, y)坐标，以米为单位，精确到毫米，用一个空格隔开x和y。

试题解析

- 给出平行四边形中两条相邻边的端点坐标，求第4个点的坐标。要注意的是，两条相邻边的端点坐标，会有两个点的坐标是重复的；因此，要判定哪两个点的坐标是重复的。
- 设给出的平行四边形中两条相邻边的端点坐标为(x_0, y_0)，(x_1, y_1)，(x_2, y_2)和(x_3, y_3)，$(x_0, y_0)= (x_3, y_3)$，求第4个点的坐标(x_a, y_b)，则有$x_a-x_2= x_1-x_0$，$y_a-y_2= y_1-y_0$；得$x_a=x_2+x_1-x_0$，$y_a= y_2+y_1-y_0$。
- 在参考C++程序中，使用交换函数swap。交换函数swap包含在命名空间std 里面，使用swap，不用担心交换变量精度的缺失，无需构造临时变量，也不会增加空间复杂度。

4.1.3 The Circumference of the Circle

- 试题来源：**Ulm Local 1996**
- 在线测试：**POJ 2242，ZOJ 1090**

- 要计算圆的周长似乎是一件容易的事，只要您知道圆的直径。但是，如果您不知道呢？给出平面上的3个非共线点的笛卡尔坐标。你的工作是计算与这3个点相交的唯一的圆的周长。

- 输入
- 输入包含一个或多个测试用例，每个测试用例一行，包含6个实数$x_1, y_1, x_2, y_2, x_3, y_3$，表示3个点的坐标。由这3个点确定的直径不超过1百万。输入以文件结束终止。
- 输出
- 对每个测试用例，输出一行，给出一个实数，表示3个点所确定圆的周长。输出的周长精确到小数两位。Pi的值为3.141592653589793。

试题解析

- 此题的关键是求出与这3个点相交的唯一圆的圆心。设3个点分别为(x_0, y_0)，(x_1, y_1)和(x_2, y_2)，圆心为 (x_m, y_m)。
- 本题采用初等几何知识解题。

试题解析

设 $a=\left|\overline{AB}\right|$，$b=\left|\overline{BC}\right|$，$c=\left|\overline{CA}\right|$，$p=\dfrac{a+b+c}{2}$

根据海伦公式 $s=\sqrt{p(p-a)(p-b)(p-c)}$、三角形面积公式 $s=\dfrac{a*b*\sin(\angle ab)}{2}$ 和正弦定理 $\dfrac{a}{\sin(\angle bc)}=\dfrac{b}{\sin(\angle ac)}=\dfrac{c}{\sin(\angle ab)}$ =外接圆直径 d，得出外接圆直径 $d=\dfrac{a*b*c}{2*s}$ 和外接圆周长 $l=d*\pi$。

4.1.4 Titanic

- 试题来源：**Ural Collegiate Programming Contest 1999**
- 在线测试：**POJ 2354，Ural 1030**

- 这是一个历史事件，在“泰坦尼克号”的传奇航程中，无线电已经接到了6封电报警告，报告了冰山的危险。每封电报都描述了冰山所在的位置。第5封警告电报被转给了船长。但那天晚上，第6封电报被延误，因为电报员没有注意到冰山的坐标已经非常接近当前船的位置了。
- 请您编写一个程序，警告电报员冰山的危险！

- 输入
- 输入电报信息的格式如下：
- Message #<n>.
- Received at <HH>:<MM>:<SS>.
- Current ship's coordinates are
- <x1>^<x2>'<x3>" <NL/SL>
- and <Y1>^<Y2>'<Y3>" <EL/WL>.
- An iceberg was noticed at
- <A1>^<A2>'<A3>" <NL/SL>
- and <B1>^<B2>'<B3>" <EL/WL>.
- ===
- 这里的<n>是一个正整数，<HH>:<MM>:<SS>是接收到电报的时间；<x1>^<x2>'<x3>" <NL/SL>和<Y1>^<Y2>'<Y3>" <EL/WL>表示"北（南）纬x1度x2分x3秒 和东（西）经Y1度Y2分Y3秒。"

- 输出
- 程序按如下格式输出消息：
- The distance to the iceberg: <s> miles.
- 其中<s>是船和冰山之间的距离（即在球面上船和冰山之间的最短路径），精确到两位小数。如果距离小于（但不等于！）100英里，程序还要输出一行文字：DANGER!

- 提示：
- 为了简化计算，假设地球是一个理想的球体，直径为6875英里，完全覆盖水。本题设定输入的每行按样例输入所显示的换行。船舶和冰山的活动范围在地理坐标上，即从0到90度的北纬/南纬（NL/SL）和从0到180度的东经/西经（EL/WL）。

试题解析

- 本题要求您计算一个球体上两点之间的距离。直接采用计算球体上距离的公式。如果距离小于100英里，则输出：“DANGER!”
- 已知球体上两点*A*和*B*的纬度和经度，分别为(wA, jA)和(wB, jB)，计算*A*和*B*之间的距离公式：$dist(A, B) = R*\arccos(\cos(wA)*\cos(wB)*\cos(jA-jB)+\sin(wA)*\sin(wB))$；其中*R*是球体的半径，默认'N'和'E'为正方向，'S'和'W'为负方向。
- 本题的输入的处理相对麻烦，先把经纬度的度、分、秒转换成度，再根据东西经，南北纬取正负号。在计算距离时，度转换为弧度；然后根据球上两点距离公式求船和冰山的距离。实现过程请参看参考程序。

4.1.5 Birthday Cake

- 试题来源： **Randy Game - Programming Contest 2001A**
- 在线测试： **UVA 10167**

- Lucy和Lily是双胞胎， 今天是她们的生日。妈妈给她们买了一个生日蛋糕。现在蛋糕被放在一个笛卡尔坐标系上，蛋糕的中心在(0, 0)，蛋糕的半径长度是100。
- 蛋糕上有2*N*（*N*为整数，1≤*N*≤50）个樱桃。妈妈要用刀把蛋糕切成两半（当然是直线）。双胞胎自然是要得到公平的对待，也就是说，蛋糕两半的形状必须相同（也就是说，直线必须穿过蛋糕的中心），而且每一半的蛋糕必须都有*N*个樱桃。您能帮她吗？
- 这里要注意，樱桃的坐标(*x*, *y*)是两个整数。您要以两个整数*A*，*B*（代表*Ax*+*By*=0）的形式给出这条直线，*A*和*B*在[−500, 500]中。樱桃不能在直线上。对于每个测试用例，至少有一个解决方案。

- 输入
- 输入包含多个测试用例。每个测试用例由两部分组成：第一部分在一行中给出一个数字*N*，第二部分由2*N*行组成，每行有两个数字，表示(*x*, *y*)。两个数字之间只有一个空格。输入以*N*=0结束。
- 输出
- 对于每个测试用例，输出一行，给出两个数字*A*和*B*，在两个数字之间有一个空格。如果有多个解决方案，则只要输出其中一个。

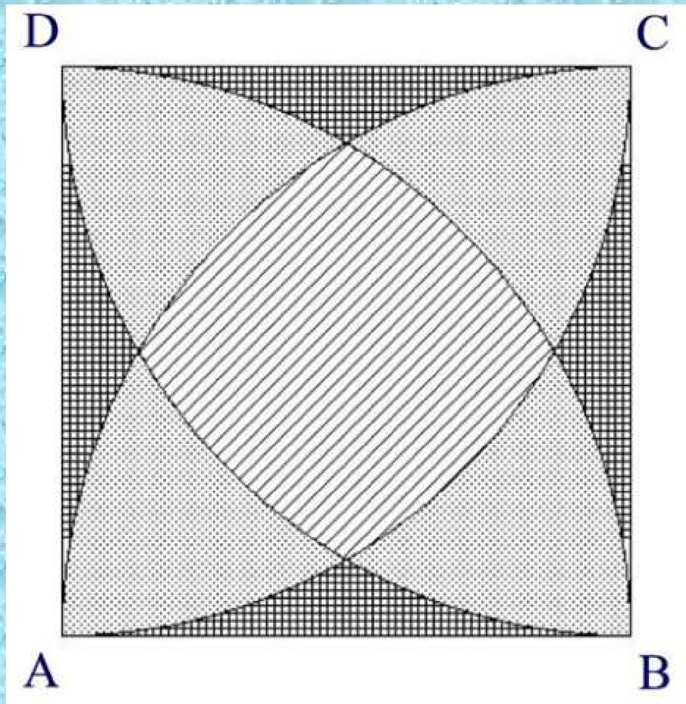
试题解析

- 本题的第一行输入*N*，表示蛋糕上有 2 *N*个樱桃。接下来的 2 *N*行每行给出一个樱桃的坐标，因为蛋糕是一个原点为圆心，半径为100的圆，所以坐标值的范围是[-100, 100]。本题的输出是一个直线方程*Ax*+*By*=0的*A*和*B*，范围是[-500, 500]。
- 本题采取枚举方法，在[-500, 500]的范围内枚举*A*和*B*，将樱桃坐标代入直线方程*Ax*+*By*，如果*Ax*+*By*大于0，则樱桃在直线上方；小于0，则樱桃在直线下方；等于0，则不允许，因为樱桃不能在直线上。枚举直至产生第一个解。

4.1.6 Is This Integration ?

- 试题来源：**Math & Number Theory Lovers' Contest 2001**
- 在线测试：**UVA 10209**

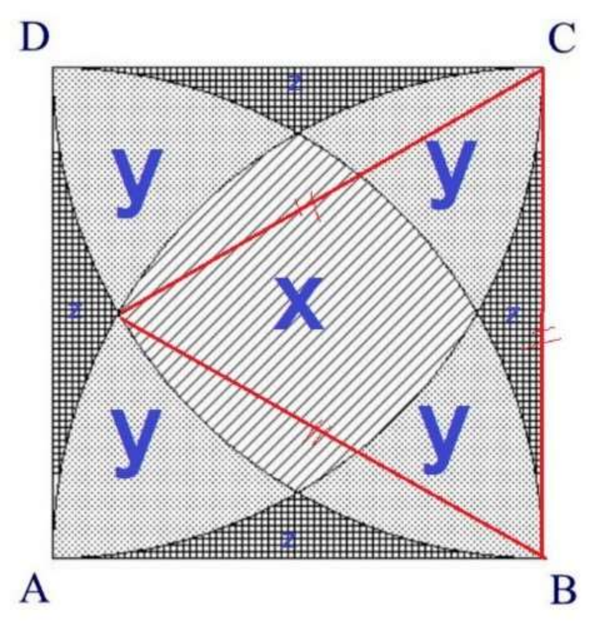
- 在图中，有一个正方形ABCD，其中，AB=BC=CD=DA=*a*。以四个顶点A, B, C, D为圆心，以*a*为半径，画四个圆弧：以A为圆心的圆弧，从相邻顶点B开始，到相邻顶点D结束；所有其他的圆弧都以类似的方式画出。如图所示，以这种方式在正方形中画出了三种不同形状的区域，每种区域用不同阴影表示。请您计算不同阴影部分的总面积。



- 输入
- 输入的每一行都给出一个浮点数a（$0 \le a \le 10000$），表示正方形的边的长度。输入以EOF结束。
- 输出
- 对于每一行的输入，输出一行，给出三种不同阴影部分的总面积：给出三个保留小数点后三位的浮点数，第一个数字表示条纹区域的总面积，第二个数字表示点星罗棋布的区域的总面积，第三个数字表示其余区域的面积。

试题解析

- 本题给出正方形的边长*a*，要求计算三种不同阴影部分的总面积。如图所示，做辅助线画一等边三角形，并且三种不同阴影部分的面积分别用*x*, *y*和*z*表示。



则 $x+4y+4z=a^2$（正方形的面积）；$x+3y+2z=\dfrac{\pi a^2}{4}$（四分之一圆面积）；而 z 的面积是正方形面积去掉等边三角形的面积和两个扇形的面积，其中扇形是正方形的边和等边三角形的边构成，而两个扇形的面积为六分之一圆面积，而等边三角形面积为 $\dfrac{\sqrt{3}}{4}a^2$，即，

$z=a^2-\dfrac{\sqrt{3}}{4}a^2-\dfrac{\pi}{6}a^2$。将 z 代入，得 $y=[(-1+\dfrac{\sqrt{3}}{2})+\dfrac{\pi}{12}]a^2$，$x=(1-\sqrt{3}+\dfrac{\pi}{3})a^2$。

4.2 欧几里得算法和扩展的欧几里得算法算法

欧几里得算法用于计算整数 a 和 b 的最大公约数（Greatest Common Divisor (GCD)）。整数 a 和 b 的最大公约数通过反复应用除运算直到余数为 0，最后的非 0 的余数就是最大公约数。欧几里得算法如下：

$$GCD(a,b)=\begin{cases} b & a=0 \\ GCD(b \bmod a, a) & \text{否则} \end{cases}=\begin{cases} a & b=0 \\ GCD(b, a \bmod b) & \text{否则} \end{cases}$$

。

【4.2.1 Simple division】是基于欧几里得算法解决问题的实验。

4.2.1 Simple division

- 试题来源： **November 2002 Monthly Contest**
- 在线测试： **UVA 10407**

- 被除数n和除数d之间的整数除运算产生商q和余数r。q是最大化$q*d$的整数，使得$q*d \le n$，并且$r=n-q*d$。
- 给出一组整数，存在一个整数d，使得每个给出的整数除以d，所得的余数相同。

- 输入
- 输入的每行给出一个由空格分隔的非零整数序列。每行的最后一个数字是0，这个0不属于这一序列。一个序列中至少有2个，至多有1000个数字；一个序列中的数字并不都是相等的。输入的最后一行给出单个0，程序不用处理该行。
- 输出
- 对于每一行输入，输出最大的整数，使得输入的每一个整数除以该数，余数相同。

试题解析

- 如果两个不同的数除以一个除数的余数相同，则这两个不同数的差值一定是除数的倍数。利用差值枚举除数即可。
- 所以，本题的算法如下：先求出原序列的一阶差分序列，然后求出所有非零元素的GCD。

给出不定方程 $ax+by=GCD(a,b)$，其中 a 和 b 是整数，扩展的欧几里得算法可以用于求解不定方程的整数根(x, y)。

设 $ax_1+by_1= GCD(a, b)$，$bx_2+(a \bmod b)y_2= GCD(b, a \bmod b)$。因为 $GCD(a, b) = GCD(b, a \bmod b)$，$ax_1+by_1=bx_2+(a \bmod b)y_2$。又因为 $a \bmod b=a-\left\lfloor \frac{a}{b} \right\rfloor *b$，$ax_1+by_1= bx_2+(a-\left\lfloor \frac{a}{b} \right\rfloor *b)y_2= ay_2+b(x_2-\left\lfloor \frac{a}{b} \right\rfloor * y_2)$。所以 $x_1=y_2$，$y_1=x_2-\left\lfloor \frac{a}{b} \right\rfloor *y_2$。因此$(x_1, y_1)$基于$(x_2, y_2)$。重复这一递归过程计算$(x_3, y_3)$，$(x_4, y_4)$，.......，直到 $b==0$，此时 $x=1$，$y=0$。

- 扩展的欧几里得算法如下。
- int *exgcd*(int *a*, int *b*, int &*x*, int &*y*)
- {
-     if (*b*==0) {*x*=1; *y*=0; return *a*;}
-     int *t*=*exgcd*(*b*, *a*%*b*, *x*, *y*);
-     int *x*0=*x*, *y*0=*y*;
-     *x*=*y*0; *y*=*x*0-(*a*/*b*)*y0;
-     return *t*;
- }

4.2.2 Euclid Problem

- 试题来源： **Sergant Pepper's Lonely Programmers Club. Junior Contest 2001**
- 在线测试： **UVA 10104**

- 由欧几里得的辗转相除法可知，对于任何正整数A和B，都存在这样的整数X和Y，$AX+BY=D$，其中D是A和B的最大公约数。本题要求，对于给定的A和B，找到对应的X，Y和D。

- 输入
- 输入给出一些行，每行由空格隔开的整数A和B组成，A, $B<1000000001$。
- 输出
- 对于每个输入行，输出一行，由三个用空格隔开的整数X、Y和D组成。如果有若干个满足条件的X和Y，那么就输出$|X|+|Y|$最小的那对。如果还是有若干个X和Y满足最小准则，则输出$X \le Y$的那一对。

试题解析

- 本题直接采用扩展的欧几里得算法进行解题。

- 【**4.2.3 Dead Fraction**】是一个基于欧几里得算法解决问题的实验。

4.2.3 Dead Fraction

- 试题来源： **Waterloo local 2003.09.27**
- 在线测试： **POJ 1930， UVA 10555**

- Mike正在拼命地要抢在最后一分钟前完成他的论文。在接下来的三天里，他要把所有的研究笔记整理成条理的形式。但不幸的是，他注意到他自己做的计算非常草率。每当他要做算术运算时，他用计算器，并把他认为有意义的答案记下来。当计算器显示了一个重复的分数时，Mike只记录前几个数字，后面跟着“...”。例如，他可能写下“0.3333...”，而不是“1/3”。但现在，他的结果需要精确的分数。然而，他没有时间重做每一次计算，所以他需要您为他编写一个自动推导原始分数的程序，而且要快！
- 本题设定，原始分数是相应于给出的数字序列的最简单的分数；也就是说，如果循环的部分有多种情形，就转化为分母最小的那一个分数。此外，本题设定Mike没有遗漏掉重要的数字；而且十进制扩展中循环部分的任何数字都没有被记录（即使循环部分都是零）。

- 输入
- 输入给出若干测试用例。对于每个测试用例，都有一行形如“0.dddd...”的输入，其中dddd是一个由1到9位数字组成的字符串，数字不能全部都为零。在最后一个测试用例后，给出包含0的一行。
- 输出
- 对于每个测试用例，输出原始分数。

- 提示：要注意到一个精确的小数有两个循环的展开式，例如，1/5 = 0.2000... = 0.19999...。

试题解析

本题要求将循环小数转化为分数，例如，0.3333...，记为0.3...，表示为分数$\frac{1}{3}$。如果循环部分有多种情形，就转化为分母最小的那一个分数。例如，0.16...，可以是最后一位6循环出现，表示为分数$\frac{1}{6}$；也可以是16循环出现，表示为分数$\frac{16}{99}$；分母最小的分数是$\frac{1}{6}$。

例如，循环小数0.3454545…，0.3是非循环部分，而此后的循环节有2位数字，组成的整数是45，则0.3454545…=0.3+0.0454545…=$\frac{3}{10}+\frac{45}{990}=\frac{3\times(100-1)+45}{990}=\frac{345-3}{990}$。

由上述实例分析，给出循环小数转化为分数的步骤。设循环小数有 n 个数字，其中，循环节有 k 个数字，在循环节前有 n-k 个非循环数字；循环小数的 n 个数字组成整数 a，n-k 个非循环数字组成整数 b。循环小数转化为分数的步骤如下：

分数的分母为 k 个 9，再补 n-k 个 0；分数的分子为 a-b；计算分母与分子的最大公因数（GCD）g；分母和分子都除以 g，化为最简分数。

例如，0.16...，最后一位 6 循环出现，所以，循环小数有 2 个数字，其中，循环节有 1 个数字，在循环节前有 1 个非循环数字；循环小数的 n 个数字组成整数 16，非循环数字组成整数 1。所以，转化为的分数是 $\frac{16-1}{90}=\frac{15}{90}=\frac{1}{6}$；如果 0.16... 是 16 循环出现，则转化为的分数是 $\frac{16}{99}$。

对于本题，以字符串输入循环小数，将“0.”之后的字符串转化为整数；然后，枚举循环节的长度，计算分子和分母；最后，输出分母最小的那一个分数。

4.3 概率论初步

- 随机现象是指这样的客观现象，当人们观察它时，所得的结果不能预先确定，而只是多种可能结果中的一种。在自然界和人类社会中，存在着大量的随机现象。掷硬币就是最常见的随机现象，可能出现硬币的正面，也可能出现硬币的反面。概率论是研究随机现象数量规律的数学分支。例如，连续多次掷一枚均匀的硬币，随着投掷次数的增加，出现正面的概率，即出现硬币正面的次数与投掷次数之比，逐渐稳定于1/2。
- 本节给出概率论的编程实验。

4.3.1 What is the Probability ?

- 试题来源： **Bangladesh 2001 Programming Contest**
- 在线测试： **UVA 10056**

- 概率一直是计算机算法中不可或缺的一部分。当确定性算法不能在短时间内解决一个问题时，就要用概率算法。在本题中，我们并不应用概率算法来解决问题。我们只是要确定某个玩家的获胜概率。
- 一个游戏是通过掷骰子一样的东西来玩的（并不设定它像普通骰子一样有六个面）。当一个玩家掷骰子时，如果某个预定情况发生（比如骰子显示3的一面朝上，绿色的一面朝上，等等），他就赢了。现在，有n个玩家。因此，先是第一个玩家掷骰子，然后第二个掷骰子，最后是第n个玩家掷骰子，再接下来，下一轮，先第一个玩家掷骰子，以此类推。当一个玩家掷骰子得到了预定的情况，他或她被宣布为赢家，比赛终止。请您确定其中一个玩家（第I个玩家）的获胜概率。

- 输入
- 输入首先给出一个整数s（$s \le 1000$），表示有多少个测试用。接下来的s行给出s个测试用例。每行先给出一个整数n（$n \le 1000$），表示玩家人数；然后给出一个浮点数字p，表示单次掷骰子时成功事件发生的概率（如果成功事件是骰子显示3的一面朝上，则p是单次掷骰子时显示3的一面朝上的概率）。对于普通骰子，显示3的一面朝上的的概率是1/6）；最后给出一个i（$i \le n$），表示要确定其获胜概率的玩家的序列号（序列号从1到n）。本题设定，在输入中，没有无效的概率（p）值。
- 输出
- 对于每一个测试用例，在一行中输出第i个玩家获胜的概率。输出浮点数在小数点后总是有四位数字，如样例输出所示。

试题解析

如果第 i 个玩家在第一轮赢，那么，在第 i 个玩家前的 i-1 个玩家都没有赢，则第 i 个玩家在第一轮赢的概率为 $(1-p)^{i-1}\times p$；同理，如果第 i 个玩家在第二轮赢，则赢的概率为 $(1-p)^{n+i-1}\times p$；以此类推，第 i 个玩家在第 k 轮赢，则赢的概率为 $(1-p)^{n(k-1)+i-1}\times p$；所以，根据加法原理和等比数列求和公式，第 i 个玩家获胜的概率计算如下：

$$
\begin{aligned}
&(1-p)^{i-1}\times p+(1-p)^{n+i-1}\times p+(1-p)^{2n+i-1}\times p+\ldots\ldots \\
&=[(1-p)^{i-1}\times p]\times[1+(1-p)^{n}+(1-p)^{2n}+\ldots\ldots] \\
&=[(1-p)^{i-1}\times p]\times\frac{1}{1-(1-p)^{n}}
\end{aligned}
$$

4.3.2 Burger

- 试题来源： **ACM Northwestern European Regionals 1996**
- 在线测试： **UVA 557**

- Clinton夫妇的双胞胎儿子Ben和Bill过10岁生日，派对在纽约南百老汇202号的麦当劳餐厅举行。派对有20个孩子参加，包括Ben和Bill。Ronald McDonald做了10个牛肉汉堡和10个芝士汉堡，当他为孩子们服务时，他先从坐在Bill左边的女孩开始，而Ben坐在Bill的右边。Ronald掷一枚硬币决定这个女孩是吃牛肉汉堡还是芝士汉堡，硬币头像的一面是牛肉汉堡，反面则是芝士汉堡。在轮到Ben和Bill之前，Ronald对其他的17个孩子也重复了这一过程。当Ronald来到Ben面前时，他就不用再掷硬币了，因为没有芝士汉堡了，只有两个牛肉汉堡。
- Ronald McDonald对此感到非常惊讶，所以他想知道这类事情发生的概率有多大。对于上述过程，请您计算Ben和Bill吃同一种汉堡的概率。Ronald McDonald总是烤制同样数量的牛肉汉堡和芝士汉堡。

试题解析

如果 Ben 和 Bill 得到不一样汉堡，也就是说，抛硬币要进行到最后，在这一过程中，每个人都要经历抛硬币决定吃哪种类型的汉堡。我们可以求 Ben 和 Bill 得到不一样汉堡的概率 p，然后 $1-p$ 即可。当派对有 $2i$ 个人时，概率 $p[i]=\dfrac{C(2i-2,i-1)}{2^{i-2}}$。

对于概率 $p[i]$，根据题目描述，数据范围是 $1 \le i \le 50000$，可以采用离线和递推来进行求解，$p[i]=1$，$p[i+1]=\dfrac{(2i-1)\times p[i]}{2i}$。

4.4 微积分初步

- 本节给出基于微积分导数的知识，编程解决问题的实验。

4.4.1 498-bis

- 试题来源： **The Joint Open Contest of Gizycko Private Higher Education Intsitute Karolex and Brest State University, 2002**
- 在线测试： **UVA 10268**

- 在“在线测试试题集文档”中，有一道非常有趣的试题，编号为498，题目名称为“Polly the Polynomial”。坦率地说，我没有去解这道试题，但我从这道试题衍生出了本题。
- 试题498的目的是“…设计这一试题是帮助你掌握基本的代数技能，等等”。本题的目的也是帮助你掌握基本的求导代数技能。
- 试题498要求计算多项式$a_0x^n + a_1x^{n-1} + . . . + a_{n-1}x + a_n$的值。
- 本题则要求计算该多项式的导数的值，对该多项式求导，得到的多项式是$a_0nx^{n-1} + a_1(n-1)x^{n-2} + . . . + a_{n-1}$。
- 本题的所有输入和输出都是整数，也就是说，其绝对值小于2^{31}。

- 输入
- 程序输入偶数行的文本。每两行为一个测试用例；其中，第一行给出一个整数，表示*x*的值；第二行则给出一个整数序列a_0, a_1, ……, a_{n-1}, a_n，表示一组多项式系数。
- 输入以EOF终止。
- 输出
- 对于每个测试用例，将给出的*x*代入求导后的多项式，并将多项式的值在一行中输出

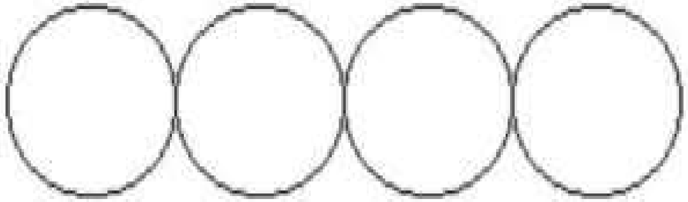
试题解析

- 本题要求计算多项式的导数值。

4.4.2 Necklace

- 在线测试：**UVA 11001**

某个部落的人用一些稀有的粘土制作直径相等的陶瓷圆环。项链由一个或多个圆环连接而成。下图显示了一条由 4 个圆环制成的项链，它的长度是每个圆盘直径的 4 倍。



每个圆环的厚度是固定的。直径 D 和粘土体积 V 具有以下关系：

$$D=\begin{cases}0.3\sqrt{V-V_0} & V>V_0\\ 0 & V\le V_0\end{cases};$$

其中，V_0是在粘土烘烤过程中被损耗掉的体积，单位和 V 一样。如果 $V \le V_0$，就不能制作陶瓷圆环。例如，如果 $V_{total}=10$，$V_0=1$。如果我们用它做一个圆环，$V=V_{total}=10$，$D=0.9$。将粘土分为两部分，每部分体积 $V=V_{total}/2=5$，则形成的每个圆环直径 $D'=0.3\sqrt{5-1}=0.6$，这样形成的项链长度为1.2。

由上面的例子可知，项链的长度随着圆环数量的变化而不同。请您编写一个程序，计算出可以做的圆环的数量，使得形成的项链是最长的。

- 输入
- 输入的每行包含两个数字，V_{total}（0< V_{total}≤60000）和V_0（0< V_0≤600），含义如上所述。输入以V_{total} = V_0=0结束。
- 输出
- 输出的每行给出可以制作圆环的数量，使得形成的项链是最长的。如果这一数字不唯一，或者根本无法形成项链，则输出“0”。

试题解析

设粘土被分成 n 份。则项链的长度为 $n \times D$，$f(n) = n \times D = n \times 0.3\sqrt{\frac{V_{total}}{n} - V_0}$。由于根号运算会有一定的误差，所以，考虑消除根号，$\frac{f(n)}{0.3} = n \times \sqrt{\frac{V_{total}}{n} - V_0}$，得新的方程式 $g(n) = \left(\frac{f(n)}{0.3}\right)^2 = n^2 \times \left(\frac{V_{total}}{n} - V_0\right) = n \times V_{total} - n^2 \times V_0$。经过上述过程，所要求的答案成为使得 $g(n)$ 最大的 n 值。

对 $g(n)$ 求导，$g'(n) = V_{total} - 2nV_0$。在导函数 $g'(n)$ 为 0 时，$g(n)$ 有极值。所以，$n = \frac{V_{total}}{2 \times V_0}$。

由于份数必须是整数，所以，在计算 n 之后，要判断最接近 n 的整数。如参考程序中所示，计算 n 和对 n 向下取整的差，如果等于 0.5，表示有两个整数解，输出 0；如果等于 0.5，则输出 n 向下取整的值；否则输出 n 向上取整的值。

4.5 矩阵计算

- 矩阵是线性代数的一个最基本的概念。本节给出矩阵的实验。
- 通常，用二维数组表示矩阵。实验【**4.5.1 Symmetric Matrix**】和【**4.5.2 Homogeneous Squares**】，方阵用二维数组表示。

4.5.1 Symmetric Matrix

- 试题来源：**Huge Easy Contest, 2007**
- 在线测试：**UVA 11349**

给出一个方阵 M，这个矩阵的元素是 M_{ij}：$\{0<i<\mathrm{n}, 0<j<n\}$。在本题中，请您判断给出的矩阵是否对称。

对称矩阵的定义：对称矩阵是这样一个矩阵，它的所有元素都是非负的，并且相对于这个矩阵的中心是对称的。任何其他矩阵都被认为是非对称的。例如：

$M=\begin{bmatrix}5 & 1 & 3\\ 2 & 0 & 2\\ 3 & 1 & 5\end{bmatrix}$是对称的，而$M=\begin{bmatrix}5 & 1 & 3\\ 2 & 0 & 2\\ 0 & 1 & 5\end{bmatrix}$则不是对称的，因为 $3\neq0$。

请您判断给出的矩阵是否对称。在输入中给出的矩阵元素为$-2^{32}\leq M_{ij}\leq2^{32}$，$0<n\leq100$。

- 输入
- 输入的第一行给出测试用例数$T \le 300$。接下来的T个测试用例按照以下方式给出。每个测试用例的第一行给出n，方阵的维数；然后给出n行，每行相应于矩阵的一行，包含n个由空格字符分隔的元素。第i行的第j个数就是矩阵的元素M_{ij}。
- 输出
- 对于每个测试用例，输出一行“Test #*t*: *S*”，其中*t*是从1开始的测试用例的编号，如果矩阵是对称的，则*S*是“Symmetric”；否则，就是“Non-symmetric”。

试题解析

- 给出的方阵用二维数组$M[100][100]$表示。对称矩阵的所有元素都是非负的，并且相对于这个矩阵的中心是对称的，所以，如果有元素是负数，或者存在相对于中心不对称，即$M[i][j] \neq M[N-1-i][N-1-j]$，则给出的方阵不是对称的。

4.5.2 Homogeneous Squares

- 试题来源： **Ulm Local 2006**
- 在线测试： **POJ 2941**

- 假设您有一个大小为n的正方形，它被划分出$n \times n$个位置，就像一个棋盘。如果存在两个位置(x_1, y_1)和(x_2, y_2)，其中$1 \le x_1, y_1, x_2, y_2 \le n$，这两个位置占据不同的行和列，即$x_1 \neq x_2$并且$y_1 \neq y_2$，则称两个位置是“独立的”。更一般地说，如果$n$个位置两两间是独立的，则称这$n$个位置是独立的。因此有$n!$种不同的选法选择$n$个独立的位置。
- 设定在这样一个$n \times n$的正方形的每个位置上都写有一个数。如果不管位置如何选择，写在n个独立位置上的数的和相等，这个正方形称为“homogeneous”。请您编写一个程序来确定一个给出的正方形是否是“homogeneous”的。

- 输入
- 输入包含若干个测试用例。
- 每个测试用例的第一行给出一个整数*n*（1≤*n*≤1000）。接下来的*n*行每行给出*n*个数字，数字之间用一个空格字符分隔。每个数字都是在区间[−1000000, 1000000]中的整数。
- 在最后一个测试用例后面跟着一个0。
- 输出
- 对于每个测试用例，按样例输出中显示的格式，输出给出的正方形是否“homogeneous”。

试题解析

本题的每个测试用例是一个 $n \times n$ 方阵，选定不同行不同列的 n 个元素，并对选定的元素求和，如果对于每一种的选法，在 n 个独立位置上的数的和相等，则输出"homogeneous"，否则输出 "not homogeneous"。

因为方阵的规模较大，$1 \le n \le 1000$，所以直接枚举肯定会超时，根据局部解递推如下的规律。

设 3×3 的方阵为$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$，其每个 2×2 的子方阵为$\begin{pmatrix} 1 & 2 \\ 4 & 5 \end{pmatrix}$，$\begin{pmatrix} 2 & 3 \\ 5 & 6 \end{pmatrix}$，$\begin{pmatrix} 4 & 5 \\ 7 & 8 \end{pmatrix}$和

$\begin{pmatrix} 5 & 6 \\ 8 & 9 \end{pmatrix}$符合 "homogeneous" 的条件，则该 3×3 的方阵是 "homogeneous" 的。

- 由这一局部解递推出规律：对于一个$n \times n$方阵，只要它的所有的$(n-1) \times (n-1)$子方阵是homogeneous的，则该$n \times n$方阵是homogeneous的；进一步递推可得，只要该$n \times n$方阵的所有的2×2的子方阵符合两对角线相加相等，该该$n \times n$方阵是homogeneous的。

- 在实验【**2.4.4 Jill Rides Again**】中，给出一个一维数组，求最大子序列和。实验【**4.5.3 To the Max**】则是给出一个二维数组，要求计算最大子矩形。

4.5.3 To the Max

- 试题来源：**ACM Greater New York 2001**
- 在线测试：**POJ 1050**

- 给出一个由正整数和负整数组成的二维数组，一个子矩形是指位于整个数组中大小为1*1或更大的任何连续子数组。矩形的和是该矩形中所有元素的和。在本题中，具有最大和的子矩形被称为最大子矩形。
- 例如，给出一个二维数组如下：
- 0 -2 -7 0
- 9 2 -6 2
- -4 1 -4 1
- -1 8 0 -2
- 最大子矩形是在左下角：
- 9 2
- -4 1
- -1 8
- 矩形的和是15。

- 输入
- 输入给出一个$N\times N$个整数组成的数组。输入的第一行给出一个正整数N，表示二维正方形数组的大小。后面给出用空白字符（空格和换行符）分隔的N^2个整数。这些整数是数组的N^2个整数，以行为顺序按行给出。也就是说，首先，第一行从左到右，给出第一行的所有数字；然后，再第二行从左到右，给出第二行的所有数字，以此类推，N的最大值可以是100。数组中的数字的范围是[-127, 127]。
- 输出
- 输出最大子矩形的和。

试题解析

本题的求解，是通过二维数组转化为一维数组，然后再求最大子序列和，以此求解最大子矩形的和。首先，通过一个实例，说明二维数组如何转化为一维数组：

设存在矩阵$\begin{pmatrix} 7 & -8 & 9 \\ -4 & 5 & 6 \\ 1 & 2 & -3 \end{pmatrix}$，将同一列中的若干个数合并。比如，从第一行开始，到第 2 行结束，每一列的和组成的序列为：3 -3 15；然后，求此序列的最大子序列和。求出后再与 *max* 比较，最后输出的一定是最大子矩形。

本题解题过程如下：

（1）第一轮：第一次仅第 1 行合并；第二次 1、2 行合并；第三次 1、2、3 行合并；……；依次类推，分别求出合并后的最大子矩形，作为局部最大值，即，包含第 1 行的最大子矩形在第一轮求出；

（2）第二轮：第一次仅第 2 行合并；第二次 2、3 行合并；第三次 2、3、4 行合并；……；依次类推，分别求出合并后的最大子矩形，作为局部最大值；即，包含第 2 行的最大子矩形在第二轮求出；

（3）第三轮：第一次仅第 3 行合并；第二次 3、4 行合并；第三次 3、4、5 行合并；……；依次类推，分别求出合并后的最大子矩形，作为局部最大值；即，包含第 3 行的的最大子矩形在第三轮求出；

……，依次类推。