2022 年常州市“程序设计小能手”比赛说明

- 文件保存的位置：在不保护硬盘分区（由监考老师当场告知）
- 文件夹命名：建立以自己准考证号为名的文件夹，例如：某学生的准考证号为 **800**，那么他的文件夹名就是 **800**，本次比赛编写的程序必须存放在这个文件夹中。
- 文件名的要求：在每题标题后的括号中有文件名，如：第一题 最大公约数（**gcd.cpp**），该题编好的程序保存时命名为 **gcd**。文件名错误不得分。
- 活动结束前将你编的所有程序(扩展名为 **cpp**)放到该文件夹中，然后右击该文件夹在弹出菜单中选择“添加到 **800.rar**”，压缩后等待工作人员上传到教师机。
- 一般说来前面的题要比后面的容易，后面的题目虽然得到满分很难，然而拿一部分分数并不难。请合理分配你的时间，先保证程序的正确性，超时等问题都是次要的，计算机的运行速度往往比你想象的要快得多。如果某题不太会做你可以针对小数据或特殊数据编程争取拿部分分数，哪怕手算一个结果输出也行，比赛总是有难度的，不能像平时学校里的小测验那样老想着拿满分，从往年的经验来看你能得到总分的 **1/3** 就相当好了。
- 所有测试点时限都是 **1** 秒，所有程序运行时内存都不能超过 **512MB**，大约可以存储六千万个 **long long** 类型的整数。
- 输出时行首和行尾都不要有多余的空格，也不要有多余的空行，相邻两项输出之间严格用一个空格隔开，一行输出结束时一定要换行。
- 所有题目均使用标准输入输出，即从键盘输入数据，结果输出到屏幕，请认真阅读范例，你的程序请严格按范例程序的格式编写。
- 题目中用到的“∧”符号表示乘方运算，如 **2^3=2×2×2=8**，**10^6=1000000**

【范例】

最大公约数和最小公倍数(gcdlcm.cpp)

<u>问题描述</u>

最大公约数(Greatest Common Divisor，简写为 GCD)：如果有一个自然数 a 能被自然数 b 整除(也称 b 能整除 a，记作 b|a)，则称 a 为 b 的倍数，b 为 a 的约数。两个自然数公共的约数，叫做这两个自然数的公约数。所有公约数中最大的一个，称为这两个自然数的最大公约数。最小公倍数(Least Common Multiple，缩写 LCM)：对于两个自然数来说，最小公倍数是指这两个数公共的倍数中最小的一个。例如： 在 12 和 16 中，4 就是 12 和 16 的最大公

约数。12 和 16 的最小公倍数是 48。

早在公元前 300 年左右，欧几里德就在他的著作《几何原本》中给出了求最小公倍数的高效方法——辗转相除法。辗转相除法使用到的原理很聪明也很简单，假设用 GCD(x, y)表示两个自然数 x 和 y 的最大公约数，取 k = x / y，b = x % y，则 x = k * y + b，如果一个自然数能够同时整除 x 和 y，则必能同时整除 y 和 b；而能够同时整除 y 和 b 的自然数也必能同时整除 x 和 y，即 x 和 y 的公约数与 y 和 b 的公约数是相同的，其最大公约数当然也相同，则当 y != 0 时有 GCD(x, y)= GCD(y, x % y)，如此便可把原问题转化为求两个更小的自然数的最大公约数，直到其中一个数为 0，剩下的另外一个数就是两者的最大公约数。以求 288 和 123 的最大公约数为例，操作如下： 288 % 123=42 123 % 42=39 42 % 39=3 39 % 3=0 所以 3 就是 288 和 123 的最大公约数。

计算最小公倍数时，通常会借助最大公约数来辅助计算。可以证明两个自然数的乘积等于它们的最大公约数和最小公倍数的乘积，即 a×b=GCD(a,b)×LCM(a,b)。如 12×16=192= GCD(12,16) ×LCM(12,16)=4×48。

编一个程序对于输入的两个自然数 a 和 b，求它们的最大公约数和最小公倍数。

输入格式

输入数据仅有一行包含两个用空格隔开的自然数 a 和 b，范围不超过 long long。

输出格式

输出数据仅有一行包含两个自然数，表示要求的最大公约数和最小公倍数。两数之间严格用一个空格隔开，行末没有多余的空格。

样例输入

12 16

样例输出

4 48

以下是 C++源程序，存盘文件名为 gcdlcm.cpp

```
#include <bits/stdc++.h>
using namespace std;
int main(){
	long long m,n,a,b,r;
	cin>>m>>n;
	a = m;
	b = n;
	while ( b != 0 ){
		r = a % b;
		a = b;
		b = r;
	}
	cout<<a<<" "<<m*n/a<<endl;
	return 0;
}
```

阶梯电价(a.cpp)

本题分值为 60 分

问题描述

碳中和是指国家、企业、产品、活动或个人在一定时间内直接或间接产生的二氧化碳或温室气体排放总量，通过植树造林、节能减排等形式，以抵消自身产生的二氧化碳或温室气体排放量，实现正负抵消，达到相对“零排放”。

龙城电力公司为了配合碳中和推出了阶梯电价的标准。以下为阶梯电价的计费说明：阶梯电价的实施周期为一年，在本年度中用电量在 2760 度以内的，算作第一阶梯，这是基本保障的用电量，每度电按 0.52 元计价；从 2761 度至 4800 度算作第二阶梯，每度电按 0.57 元计价，电价与发电平均成本持平；第三阶梯从 4801 度开始计费，每度电按 0.82 元计价，以促进节约用电。每过一年电表读数归零，每个月的电费视用电情况收取，每度电的价格取决于这度电属于哪个阶梯。

小 X 有个每月底查看自家电表的习惯，并记录下每个月电表的读数。现在小 X 知道了上个月底的电表读数 u 和当前这个月底的电表读数 v(其中 u≤v，且 u 和 v 都是大于等于 0 的整数即非负整数)，小 X 想知道按照阶梯电价的标准，当前这个月的电费是多少，你能写一个程序帮他计算一下吗？

输入格式

一行用空格隔开的两个整数 u 和 v，分别表示上个月底的电表读数和当前这个月底的电表读数。

输出格式

一行一个实数，表示当前这个月的电费，要求输出时保留 2 位小数。

样例输入

2700 5000

样例输出

1358.00

样例解释

小X家上个月底的电表读数为2700，表示到上个月结束为止已经用掉了本年度第一阶梯的2700度电，这个月先用掉了第一阶梯余下的60度电，然后开始进入第二阶梯，并将第二阶梯的额度共计2040度电全部用完，紧接着继续用掉了第三阶梯的200度电，所以这个月的总电费为：60*0.52+2040*0.57+200*0.82=1358.00元。

数据规模与约定

测试点编号	u	v
1	u≤2760	2760＜v≤4800
2	u≤2760	v＞4800

3	2760＜u≤4800	v＞4800
4	0≤u≤v≤2760	
5	2760＜u≤v≤4800	
6	4800＜u≤v≤10000	

最早对决 (b.cpp)

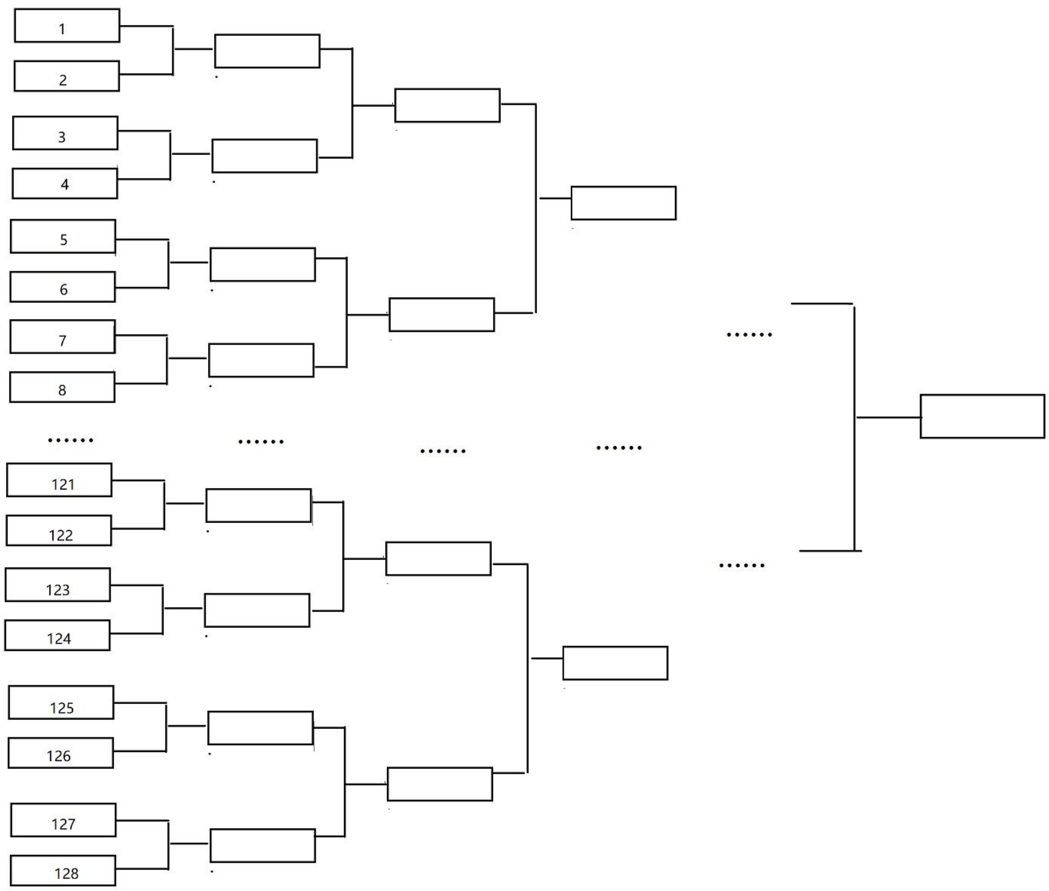
本题分值为 70 分

<u>**问题描述**</u>

小X和小Y凭借出色的程序设计能力，顺利入选了“我们爱科学”活动，该活动以科学精神、创新意识、实践能力的培养为目标，激发广大中小学生爱科学、学科学、用科学的兴趣。

报到之后小X和小Y有幸被分进了人工智能(简称AI)班，经过一周紧张刺激的AI学习，每位同学都编写了一个AI程序“黑白棋”，经过对弈平台的双循环赛，一共选出了32位种子选手，编号分别为1～32，其余选手均为非种子选手，编号为33～128。

最后的决战拉开帷幕，比赛采用淘汰赛制，共有128位选手参加这场淘汰赛，两两对决，胜者进入下一轮，而输的将被直接淘汰出局。通过抽签产生这128位选手的对阵表，抽签规则如下：



首先将种子选手的编号从小到大排序，1号种子安排在1号位置，2号种子安排在128号位置。接着将整个赛区分为上半赛区和下半赛区，即上半赛区是1～64号位置，下半赛区是65～128号位置，将上半赛区的最后一个位置和下半赛区的第一个位置选出来(64号和65号)，然后将接下来的2位种子选手(3号和4号)通过随机抽签的方式安排在这2个位置中，也就是说3号种子既可能抽到64号位置，也可能抽到65号位置，3号种子的位置通过抽签定下来之后，4号种子就自动被安排到另一个位置；接下去对于每个赛区再继续分上半赛区和下半赛区，这样就形成了1～32，33～64，65～96，97～128四个赛区，将所有上半赛区的最后一个位置和下半赛区的第一个位置选出来(32号,33号,96号,97号)。然后将接下来的4位种子选手(5号,6号,7号,8号)通过随机抽签的方式安排到这4个位置中，然后继续划分赛区，直到所有种子选手被安排完毕，剩下的96个非种子选手通过随机抽签安排到余下空位中。

小X和小Y的AI十分强大，两人的AI可以打败除了他们以外的所有对手，但是他们发现，假如小X的编号是1号，小Y的编号是2号，他们将作为头两号种子坐在1号位置和128号位置，两人将所向披靡一路击败所有的对手，直到决赛(第7轮)才会相遇。

现在给出小X和小Y的编号，他们想知道最早会在哪一轮相遇？

输入格式

输入数据仅有一行包含两个用空格隔开的正整数S和T，表示小X和小Y的编号。

输出格式

输出数据仅有一行包含一个正整数，表示他们最早相遇的轮数。

第1轮：128进64

第2轮：64进32

第3轮：32进16

第4轮：16进8

第5轮：8进4

第6轮：半决赛

第7轮：决赛

样例输入

1 3

样例输出

6

样例解释

小 X 是 1 号种子，坐在 1 号位置，小 Y 是 3 号种子，他可能抽到 64 号或 65 号位置，如果小 Y 抽到 64 号位置，他跟小 X 会在半决赛(第 6 轮)相遇。如果小 Y 抽到 65 号位置，他跟小 X 会在决赛(第 7 轮)相遇。所以他们最早会在第 6 轮相遇。

数据规模与约定

对于10%的数据，小X和小Y都是非种子选手即编号都大于32

对于另外10%的数据，小X和小Y有一个是非种子选手

对于100%的数据，1≤S,T≤128

可能的三角形 (c.cpp)

本题分值为 80 分

问题描述

小 X 和小 Y 都是龙城学堂的资深学员，为了丰富学弟学妹们的课余生活，小 X 和小 Y 发明了一个简单的数字游戏。小 Y 有三个正整数 A,B,C(2≤A≤B≤C)，且 A,B,C 刚好构成一个三角形的三条边。这些数字是保密的，他不会直接透露给小 X。他会告诉小 X 一共 4 个正整数 x1,x2,x3,x4，其中 2≤x1,x2,x3,x4≤10^9，并宣称 x1,x2,x3,x4 每个数必定是 A,B,C,A+B,A+C,B+C 或 A+B+C 之一。为了公平起见，小 Y 不可能撒谎，也就是说他给出的这些正整数里一定存在至少一组对应的合法的(A,B,C)，满足 A,B,C 恰好是某个三角形的三条边。

小 X 百思不得其解，所以请你来求出有哪些三元组(A,B,C)符合条件。

输入格式

一行四个用空格隔开的正整数 x1,x2,x3,x4。

输出格式

输出若干行，每行三个整数，两数之间严格用一个空格隔开，表示一组可能的(A,B,C)，使得 A,B,C 恰好是某个三角形的三条边。输出时要求按照 A 升序输出，如果 A 相同则按照 B 升序输出，如果 A,B 都相同则按照 C 升序输出。所谓升序是指从小到大的次序，输入数据保证至少有一组解。

样例输入

2 4 5 7

样例输出

2 2 3
2 3 4
2 4 5

样例解释

对于第一组解，A=2，B=2，C=3，输入的4个数对应的值分别是A,A+B,A+C,A+B+C；

对于第二组解，A=2，B=3，C=4，输入的4个数对应的值分别是A,C,A+B,B+C；

对于第三组解，A=2，B=4，C=5，输入的4个数对应的值分别是A,B,C,A+C；

除此之外不可能存在其它符合条件的三角形了。注意(1,2,4)不可能组成三角形。

数据规模与约定

测试点编号	特殊性质
1	保证答案的A,B,C构成直角三角形
2	保证答案的A,B,C构成等腰三角形
3	保证答案的A,B,C构成等边三角形
1～6	保证答案的A,B,C唯一

7～10	1≤x1,x2,x3,x4≤100
11～20	1≤x1,x2,x3,x4≤10^9

文本找数（d.cpp）

本题分值为 90 分

问题描述

正当小 X 带着大家玩三角形游戏的时候，小 Y 在编程中遇到了一个难题，来请教小 X。小 Y 遇到的难题是需要在一行文本中找最大的数，这一行文本中有整数和实数，也有字母、空格等其它各种字符。这一行文本最多包括 10000 个字符，其中每个数的长度不超过 100(包括小数点)，所有的数都没有正负符号，并且没有前导 0，所谓前导 0 是指一个数中开头可以省略的 0，如 007 中的两个 0 就是前导 0，0.618 中小数点前的 0 不是前导 0。也就是说文本中出现的整数必定是一串连续的阿拉伯数字，除了 0 以外所有整数的首位一定不是 0。文本中出现的实数必定是一串连续的阿拉伯数字并且中间夹杂着一个唯一的小数点，该小数点两侧必须是数字，并且小数点左侧一定是上文所述的整数。

输入格式

输入数据仅有一行包含一个字符串，表示要查找的文本。保证文本中小数点的前后位置一定都是数字，并且不存在一串连续的阿拉伯数字的前后各有一个小数点的情况，像“120.78.90.206”这样的字符串是不可能出现在文本中的。

输出格式

一行一个整数或实数(按照文本中原来的样子原封不动输出)。如果最大的数不至一个，则输出长度最长的那个数。

样例输入 1

120 315 513 512 153 0

样例输出 1

513

样例输入 2

5r2.1q 4p 3.77442qw cock5.0$

样例输出 2

5.0

样例解释

对于样例1，有120、315、513、512、153、0六个数，最大的数为513。

对于样例2，有5、2.1、4、3.77442、5.0五个数，最大的数有两个，5.0的长度更长。

数据规模与约定

对于10%的数据，文本为用空格隔开的若干个非负整数(范围在int以内)，且最后一个数

为0，其余的数都不为0

对于另外40%的数据，文本中不包含小数点

对于 100%的数据，文本长度≤10000，单个数的长度≤100

青蛙游泳（e.cpp）

本题分值为 100 分

问题描述

忙碌了一天，即将夕阳西下，小 X 决定趁着天气宜人到附近的郊外走一走，他在那里看到池塘中有青蛙在游泳，这些青蛙触发了小 X 出题的灵感。

池塘总长度为 L，现在有两只青蛙同时从两岸下水，第一只青蛙从左岸下水向右岸游，速度为 V1，第二只青蛙从右岸下水向左岸游，速度为 V2。当青蛙游到对岸时会改变方向折返往回游，直到游完规定时间。假设青蛙改变方向折返不需要消耗时间且保持速度不变，池塘的两岸是平行的，青蛙游泳的方向是一条直线并且垂直于两岸，小 X 想知道在 T 个单位时间内两只青蛙会相遇几次。你能编写个程序帮他计算一下吗？

输入格式

输入数据仅有一行包含四个用空格隔开的整数 L,V1,V2,T。其中 L 和 T 为正整数，V1 和 V2 为非负整数。

输出格式

一行一个整数表示答案。

样例输入1

5 1 0 6

样例输出1

1

样例输入2

6 3 3 10

样例输出2

5

样例解释

对于样例1，第一只青蛙速度为1，第二只青蛙速度为0，表示它跳下水后没有移动，池塘长度为5，在6个单位时间内能相遇1次，相遇时间发生在第5个单位时间结束时，相遇地点为池塘的右岸。

对于样例2，两只青蛙速度相同，均为3，池塘长度为6，在10个单位时间内，两只青蛙各游了5个单程，每个单程都会在池塘正中迎面相遇，然后继续游到对岸后折返，总共会相遇5次。

数据规模与约定

测试点编号	L,V2,V2,T≤	特殊性质1	特殊性质2
1	10^3	V1=V2	T时间结束时，保证两只青蛙都在池塘的端点处。
2		V2是V1的倍数	
3		/	
4	10^9	V1=V2	
5		V2是V1的倍数	
6		/	
7		V1=V2	/
8		V2是V1的倍数	
9		/	
10			

迷宫探险 (f.cpp)

本题分值为 100 分

问题描述

完成了俱乐部可人老师布置的命题任务，小 X 决定和朋友们玩一款探险类游戏放松一下心情。这个游戏的场景是在一个地下迷宫中，这个迷宫由 N*N 的网格构成，小 X 和他的朋友们每人占据一个格子，他们每人带领一支探险队。每一分钟小 X 会让他的探险队员从上下左右四个方向前往相邻的格子(只要相邻的格子不是障碍物)，同时他的朋友们也会跟小 X 一样做相同的操作。迷宫中有些格子是空的，而有些格子有自动计分器，最早到达这个格子的队员所属的玩家会得到 1 分，然后这个自动计分器会消失，即之后到达这个格子就不会获得分数，如果有多个玩家的队员同时到达有自动计分器的格子，那么这些玩家都能得到 1 分。现在小 X 想知道得分最多的玩家得到了多少分，以及所有玩家一共得到了多少分。你可以认为每位玩家的手下都有足够多的探险队员。

输入格式

第一行一个整数 N，表示地下迷宫的大小。

接下来 N 行，每行 N 个字符，‘.’表示这个格子是个空地，‘#’表示这个格子是个障碍物，‘@’表示开始时有玩家在这个格子，‘$’表示这个格子有自动计分器。除了障碍物所在格子，所有格子均可通行。

输出格式

输出数据共有两行，每行一个整数，第一行的整数表示得分最多的玩家得到了多少分，第二行的整数表示所有的玩家一共得到了多少分。

样例输入

3

@$#
#$#
#$@

样例输出

2
4

样例解释

在第一分钟，位于(1,1)的玩家派出的探险队员到达(1,2)得到1分，位于(3,3)的玩家派出的探险队员到达(3,2)得到1分。在第二分钟，两位玩家派出的探险队员同时到达(2,2)，各得1分，之后即使游戏再进行下去也没有意义了，因为他们不可能再得到任何分数。得分最多的玩家得到的分数为2分，所有玩家一共得到了4分。

数据规模与约定

测试点编号	N	特殊性质
1	≤10	只有一个玩家
2	≤10	只有一个得分点
3～5	≤10	无
6～8	≤50	
9～10	≤100	

对于100%的数据，玩家的数量≤10。

均分纸牌(g.cpp)

本题分值为 100 分

问题描述

经历了忙碌而充实的一天，小 X 正准备上床睡觉，这时他看到书桌上有一些纸牌被分成了 n 堆，n 堆纸牌排成一行，编号为 1,2,⋯,n，每堆纸牌有一定的张数(张数可能为 0，第 i 堆的张数记为 ai)。见此情景，小 X 脑海中瞬间浮现出一道经典的编程题《均分纸牌》，他觉得如果在原题的基础上修改一些条件，将是一道非常好的压轴题。于是小 X 立刻拿出了纸和笔，认真地思考起来，首先他把全部纸牌的总张数改为不必为 n 的倍数，其次他将移动规则和最终目标也作了调整，移动规则改为可以在任意两堆之间移动任意张纸牌，目标是让张数最多的那堆纸牌的张数与张数最少的那堆纸牌的张数的差≤1。已知将第 i 堆的一张纸牌移动到第 j 堆的代价为|i-j|，|i-j|的值等于 i 与 j 的差值，如 i=3,j=5 时，|i-j|等于 2，反之 i=5,j=3 时，|i-j|还是等于 2，也就是说无论你从第 3 堆向第 5 堆还是从第 5 堆向第 3 堆移动 1 张纸牌，所需的代价均为 2。现在小 X 想知道为了达成目标，他所消耗的代价最小为多少？

例如，当 n=5 时：

堆号 1　2　3　4　5

张数 5　9　2　12　9

移动的方法有多种，　其中的一种方案：

① 第 2 堆向第 1 堆移动 2 张，成为：7　7　2　12　9，消耗代价为 1*2=2

② 第 4 堆向第 3 堆移动 4 张，成为：7 7 6 8 9，消耗代价为 1*4=4

③ 第 5 堆向第 3 堆移动 1 张，成为：7 7 7 8 8，消耗代价为 2*1=2

张数最多的堆有 8 张纸牌，张数最少的堆为 7 张纸牌，达成了任意两堆纸牌的张数差≤1 的目标，此时付出的代价为 2+4+2=8，可以证明找不到更小的可以达成目标的代价。

输入格式

第一行一个整数 n，表示纸牌的堆数。

第二行有 n 个用空格隔开的非负整数，表示每堆纸牌的张数。

输出格式

一行一个整数，表示所消耗的最小代价。

样例输入

5
5 9 2 12 9

样例输出

8

数据规模与约定

对于20%的数据，n≤10，ai≤10
对于另外30的数据，保证纸牌的总数一定是n的倍数
对于100%的数据，n≤1000，ai≤10^6