实验三

**1.小希的数表**

【问题描述】

Gardon 昨天给小希布置了一道作业，即根据一张由不超过 5000 的 N(3<=N<=100)个正整数组成的数表两两相加得到 N\*(N-1)/2 个和，然后再将它们排序。例如，如果数表里含有四个数 1，3，4，9，那么正确答案是 4，5，7，10，12，13。小希做完作业以后出去玩了一阵，可是下午回家时发现原来的那张数表不见了，好在她做出的答案还在，你能帮助她根据她的答案计算出原来的数表么？

【输入形式】

包含多组数据，每组数据以一个 N 开头，接下来的一行有按照大小顺序排列的 N\*(N-1)/2 个数，是小希完成的答案。文件最后以一个 0 结束。

假设输入保证解的存在性和唯一性。

【输出形式】

对于每组数据，输出原来的数表。它们也应当是按照顺序排列的。

【样例输入】

4

4 5 7 10 12 13

4

5 6 7 8 9 10

0

【样例输出】

1 3 4 9

2 3 4 6

**2.变形课**

【问题描述】

呃……变形课上Harry碰到了一点小麻烦,因为他并不像Hermione那样能够记住所有的咒语而随意的将一个棒球变成刺猬什么的,但是他发现了变形咒语的一个统一规律:如果咒语是以a开头b结尾的一个单词,那么它的作用就恰好是使A物体变成B物体.

Harry已经将他所会的所有咒语都列成了一个表,他想让你帮忙计算一下他是否能完成老师的作业,将一个B(ball)变成一个M(Mouse),你知道,如果他自己不能完成的话,他就只好向Hermione请教,并且被迫听一大堆好好学习的道理。

【输入形式】

测试数据有多组。每组有多行，每行一个单词,仅包括小写字母,是Harry所会的所有咒语.数字0表示一组输入结束.

【输出形式】

如果Harry可以完成他的作业,就输出”Yes.”,否则就输出”No.”(不要忽略了句号)

【输入样例】

so  
soon  
river  
goes  
them  
got  
moon  
begin  
big  
0

【输出样例】

Yes.

**3. Gizilch**

【问题描述】

The game of gizilch has very simple rules. First 100 grapes are labeled, in nontoxic ink, with the numbers 1 to 100. Then, with a cry of ``GIZILCH!'', the referee fires the grapes up into the air with a giant gizilcher. The two players, who each start with a score of ``1'', race to eat the falling (or, shortly thereafter, fallen) grapes and, at the same time, multiply their scores by the numbers written on the grapes they eat. After a minute, the hungry squirrels are let loose to finish the remaining grapes, and each contestant reports his score, the product of the numbers on the grapes he's eaten. The unofficial winner is the player who announces the highest score.  
Inevitably, though, disputes arise, and so the official winner is not determined until the disputes are resolved. The player who claims the lower score is entitled to challenge his opponent's score. The player with the lower score is presumed to have told the truth, because if he were to lie about his score, he would surely come up with a bigger better lie. The challenge is upheld if the player with the higher score has a score that cannot be achieved with grapes not eaten by the challenging player. So, if the challenge is successful, the player claiming the lower score wins.  
So, for example, if one player claims 343 points and the other claims 49, then clearly the first player is lying; the only way to score 343 is by eating grapes labeled 7 and 49, and the only way to score 49 is by eating a grape labeled 49. Since each of two scores requires eating the grape labeled 49, the one claiming 343 points is presumed to be lying.  
On the other hand, if one player claims 162 points and the other claims 81, it is possible for both to be telling the truth (e.g. one eats grapes 2, 3 and 27, while the other eats grape 81), so the challenge would not be upheld.  
Unfortunately, anyone who is willing to referee a game of gizilch is likely to have himself consumed so many grapes (in a liquid form) that he or she could not reasonably be expected to perform the intricate calculations that refereeing requires. Hence the need for you, sober programmer, to provide a software solution.

【输入形式】

Pairs of unequal, positive numbers, with each pair on a single line, that are claimed scores from a game of gizilch.

【输出形式】

Numbers, one to a line, that are the winning scores, assuming that the player with the lower score always challenges the outcome.

【输入样例】

343 49

3599 610

62 36

【输出样例】

49

610

62

**4.Tempter of the Bone**

【问题描述】

The doggie found a bone in an ancient maze, which fascinated him a lot. However, when he picked it up, the maze began to shake, and the doggie could feel the ground sinking. He realized that the bone was a trap, and he tried desperately to get out of this maze.  
  
The maze was a rectangle with sizes N by M. There was a door in the maze. At the beginning, the door was closed and it would open at the T-th second for a short period of time (less than 1 second). Therefore the doggie had to arrive at the door on exactly the T-th second. In every second, he could move one block to one of the upper, lower, left and right neighboring blocks. Once he entered a block, the ground of this block would start to sink and disappear in the next second. He could not stay at one block for more than one second, nor could he move into a visited block. Can the poor doggie survive? Please help him.

【输入形式】

The input consists of multiple test cases. The first line of each test case contains three integers N, M, and T (1 < N, M < 7; 0 < T < 50), which denote the sizes of the maze and the time at which the door will open, respectively. The next N lines give the maze layout, with each line containing M characters. A character is one of the following:  
  
'X': a block of wall, which the doggie cannot enter;  
'S': the start point of the doggie;  
'D': the Door; or  
'.': an empty block.  
  
The input is terminated with three 0's. This test case is not to be processed.

【输出形式】

For each test case, print in one line "YES" if the doggie can survive, or "NO" otherwise.

【输入样例】

4 4 5

S.X.

..X.

..XD

....

3 4 5

S.X.

..X.

...D

0 0 0

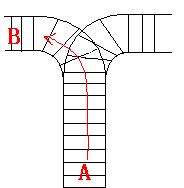
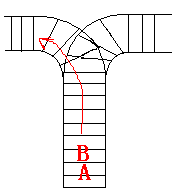
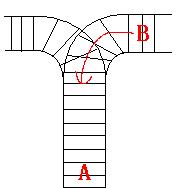
【输出样例】

NO

YES

**5. Train Problem I**

Problem Description

As the new term comes, the Ignatius Train Station is very busy nowadays. A lot of student want to get back to school by train(because the trains in the Ignatius Train Station is the fastest all over the world ^v^). But here comes a problem, there is only one railway where all the trains stop. So all the trains come in from one side and get out from the other side. For this problem, if train A gets into the railway first, and then train B gets into the railway before train A leaves, train A can't leave until train B leaves. The pictures below figure out the problem. Now the problem for you is, there are at most 9 trains in the station, all the trains has an ID(numbered from 1 to n), the trains get into the railway in an order O1, your task is to determine whether the trains can get out in an order O2.  


Input

The input contains several test cases. Each test case consists of an integer, the number of trains, and two strings, the order of the trains come in:O1, and the order of the trains leave:O2. The input is terminated by the end of file. More details in the Sample Input.

Output

The output contains a string "No." if you can't exchange O2 to O1, or you should output a line contains "Yes.", and then output your way in exchanging the order(you should output "in" for a train getting into the railway, and "out" for a train getting out of the railway). Print a line contains "FINISH" after each test case. More details in the Sample Output.

Sample Input

3 123 321

3 123 312

Sample Output

Yes.

in

in

in

out

out

out

FINISH

No.

FINISH

**6.组合总和**

【问题描述】

给你一个 无重复元素 的整数数组 candidates 和一个目标整数 target ，找出 candidates 中可以使数字和为目标数 target 的 所有 不同组合 ，并以列表形式返回。你可以按 任意顺序 返回这些组合。

candidates 中的 同一个 数字可以 无限制重复被选取 。如果至少一个数字的被选数量不同，则两种组合是不同的。

对于给定的输入，保证和为 target 的不同组合数少于 150 个。

【示例1】

输入：candidates = [2,3,6,7], target = 7

输出：[[2,2,3],[7]]

解释：

2 和 3 可以形成一组候选，2 + 2 + 3 = 7 。注意 2 可以使用多次。

7 也是一个候选， 7 = 7 。

仅有这两种组合。

【示例1】

输入: candidates = [2,3,5], target = 8

输出: [[2,2,2,2],[2,3,3],[3,5]]

示例 3：

输入: candidates = [2], target = 1

输出: []

提示：

1 <= candidates.length <= 30

2 <= candidates[i] <= 40

candidates 的所有元素 互不相同

1 <= target <= 40

# 7. N皇后问题

在N\*N的方格棋盘放置了N个皇后，使得它们不相互攻击（即任意2个皇后不允许处在同一排，同一列，也不允许处在与棋盘边框成45角的斜线上。你的任务是，对于给定的N，求出有多少种合法的放置方法。

Input

共有若干行，每行一个正整数N≤10，表示棋盘和皇后的数量；如果N=0，表示结束。

Output

共有若干行，每行一个正整数，表示对应输入行的皇后的不同放置数量。

Sample Input

1

8

5

0

Sample Output

1

92

10

**8.八球胜负问题**

【问题描述】

8球是一种台球竞赛的规则。台面上有7个红球、7个黄球以及一个黑球，当然还有一个白球。对于本题，我们使用如下的简化规则：红、黄两名选手轮流用白球击打各自颜色的球，如果将该颜色的7个球全部打进，则这名选手可以打黑球，如果打进则算他胜。如果在打进自己颜色的所有球之前就把黑球打进，则算输。如果选手不慎打进了对手的球，入球依然有效。

现在给出打进的球（白球除外）的顺序，以及黑球由哪方打进，你的任务是判定哪方是胜者。

假设不会有一杆同时打进一颗黑球和其他彩球。

【输入形式】

输入包含多组数据。每组数据第一行是一个整数NN(1≤N≤15)，表示打进的球的个数，N=0表示结束。随后有一行，包含N个字符，依序表示打进的是何种球。如果是B，表示是红方打进的黑球，如果是L，表示是黄方打进的黑球。如果是Y则表示是黄球，R表示红球。字符间没有空格。

所有输入都满足如下条件：最后一颗球打进时这局比赛正好结束，而且打进的红球和黑球都不超过7个。

【输出形式】

对每组数据，输出一行。如果红方胜，输出Red；黄方胜，输出Yellow。

【输入样例】

5  
RYRRB  
9  
RRRRYRRRB  
0

【输出样例】

Yellow  
Red

**9.DFS**

【问题描述】

A DFS(digital factorial sum) number is found by summing the factorial of every digit of a positive integer.

For example ,consider the positive integer 145 = 1!+4!+5!, so it's a DFS number.  
  
Now you should find out all the DFS numbers in the range of int( [1, 2147483647] ).  
  
There is no input for this problem. Output all the DFS numbers in increasing order. The first 2 lines of the output are shown below.

【输入样式】

no input

【输出样式】

Output all the DFS number in increasing order.

【输出样例】

1

2

......