Problem A: Burkina Faso

(blue balloon)

There are at least three telecom operators in Burkina Faso [Airtel, Onatel (Office National des Telecommunications) and Telecel Faso]. Each operator has different prices for call and data usage, given in the table below. All prices are in XOF (FCFA) :

#	Name	Call (per minute)	Data (per megabyte)
1	Airtel	30	40
2	Onatel	35	30
3	Telecel	40	20

Some foreign students have arrived Burkina Faso to participate in the Marathon Challenge, Ouagadougou Site. They already know how many minutes they will call, and how much Internet they will use. For each student, you want to recommend an operator to minimize the total cost of call usage and data usage for that student.

Standard Input

Each line of the input contains the information of one student. For each student, there are two positive integers c and d ($1 \leqslant c, d \leqslant 1000$) that show the amount of call (in minutes) and data usage (in megabytes) for the student, respectively. The input terminates with "`0 0`" that should not be processed.

Standard Output

For each student, print a line containing the minimum total cost of call usage and data usage.

Sample Input	Sample Output
10 60 100 20 24 12 900 400 50 50 0 0	1600 3800 1200 43000 3000

Problem B: Payment card

(red balloon)

An android market, is looking for creative software developers. A group of applicants are attending an interview, and the company wants to select the fastest developer who can code simple rules accurately. As a test, all applicants should quickly develop a bank card verifier that determines whether a payment card number is valid or not.

All payment card numbers are 16 digits long. The leftmost 6 digits represent a unique identification number for the bank who has issued the card. The next 2 digits determine the type of the card (e.g., debit, credit, gift). Digits 9 to 15 are the serial number of the card, and the last digit is used as a control digit to verify whether the card number is valid. Hence, if somebody enters the card number incorrectly, there is a high chance that a payment software can easily determine it.

For a valid card number, the last digit is selected in such a way that the following algorithm passes:

1. Label all digits from left to right by 1 to 16.
2. Multiply each odd-labeled digit by 2.
3. If the result for any digit is greater than 9, subtract 9 from it.
4. Sum the results of the previous step, and add to it the sum of all even-labeled digits.
5. If the result is a multiple of 10, the card number is valid; otherwise, it is invalid.

Your task is to read several card numbers from the input, and determine whether each one is a valid card number or not.

Standard Input

There are multiple test cases in the input. Each test is given in one line consisting of four space-separated 4-digit strings. The leftmost digit of the given card number is guaranteed to be non-zero. The input terminates with a line containing "`0000 0000 0000 0000`" that should not be processed.

Standard Output

For each test case, output a line containing "`Yes`" or "`No`" depending on whether the card number is valid or not, respectively.

Sample Input	Sample Output
6104 3376 7866 1545 6104 3376 7866 1546 5022 2910 0140 7954 0000 0000 0000 0000	Yes No Yes

Problem C: FIFA

(yellow balloon)

The FIFA has released the draw procedure for the FIFA World Cup on the official FIFA website, explained below.

The 32 qualified finalists are first distributed into four seeding pots based on the FIFA ranking. Pot 1 contains the host Russia and the seven highest ranking teams; the next eight highest ranked teams in Pot 2, and so on. Then, teams are drawn into eight groups of four, which are labeled from A to H. The pots are emptied into groups in order from Pot 1 through Pot 4. The draw must satisfy the following two rules:

(i) no teams from the same pot can be drawn into the same group.

(ii) with the exception of UEFA, which has more qualified teams (14) than the groups (8), no teams from the same confederation can be drawn in the same group. Moreover, at most two teams from UEFA can be drawn in the same group.

Order groups alphabetically from A to H with A and H respectively being the leftmost and rightmost groups. At each step of the draw, the drawn team x from Pot i is placed in the first group from left (starting from group A) not violating rules (i) and (ii) and being possible to distribute the remaining teams (not drawn teams so far) of Pot i in the next steps without violating rules (i) and (ii). Computer scientists have assured FIFA that it is always possible to distribute teams of Pot i into groups satisfying rules (i) and (ii), regardless of how the other teams in Pot 1 through Pot i - 1 are distributed into groups.

The table below shows the pots. In this table, x (r, c) means that team x belongs to confederation c, and its rank in the FIFA ranking is r. You are to write a program to simulate the draw and report the groups for the given draw order for each pot.

Seeding Pot 1	Seeding Pot 2	Seeding Pot 3	Seeding Pot 4
Russia (65, UEFA)	Spain (8, UEFA)	Denmark (19, UEFA)	Serbia (38, UEFA)
Germany (1, UEFA)	Peru (10, CONMEBOL)	Iceland (21, UEFA)	Nigeria (41, CAF)
Brazil (2, CONMEBOL)	Switzerland (11, UEFA)	Costa Rica (22, CONCACAF)	Australia (43, AFC)
Portugal (3, UEFA)	England (12, UEFA)	Sweden (25, UEFA)	Japan (44, AFC)
Argentina (4, CONMEBOL)	Colombia (13, CONMEBOL)	Tunisia (28, CAF)	Morocco (48, CAF)
Belgium (5, UEFA)	Mexico (16, CONCACAF)	Egypt (30, CAF)	Panama (49, CONCACAF)
Poland (6, UEFA)	Uruguay (17, CONMEBOL)	Senegal (32, CAF)	South Korea (62, AFC)
France (7, UEFA)	Croatia (18, UEFA)	Iran (34, AFC)	Saudi Arabia (63, AFC)

Standard Input

There are multiple test cases in the input. Each test case consists of 4 lines. The ith line presents all 8 team names (as written in the table) in Pot i in the draw order (from left to right). Team names are separated by ”,” and there may exist a space before or after team names. In all test cases, Russia is the first drawn country in Pot 1 due to the old tradition of placing the host country in group A. The input terminates with “`End`” that should not be processed.

Standard Output

For each test case, simulate the draw based on the given draw order, and compute the weight of each group which is the summation of team ranks in that group. For each group, print the group name (an uppercase letter) and its weight separated by a space in one line. This must be done in the increasing order of the weights from the strongest group (the group with the minimum weight) to the weakest group (the group with the maximum weight). In the case of a tie, print based on the alphabetical order of group names.

Sample Input	Sample Output
Russia, Germany, Brazil, Portugal, Argentina, Belgium, Poland, France Spain, Peru, Switzerland, England, Colombia, Mexico, Uruguay, Croatia Denmark, Iceland, Costa Rica, Sweden, Tunisia, Egypt, Senegal, Iran Serbia, Nigeria, Australia, Japan, Morocco, Panama, South Korea, Saudi Arabia Russia, Germany, Brazil, Portugal, Argentina, Belgium, Poland, France Spain, Peru, Switzerland, England, Colombia, Mexico, Uruguay, Croatia Denmark, Iceland, Costa Rica, Sweden, Tunisia, Egypt, Senegal, Iran Serbia, Nigeria, Morocco, Panama, Australia, Japan, South Korea, Saudi Arabia End	B 73 C 78 E 83 D 92 H 107 F 110 G 118 A 136 C 77 B 78 E 83 D 87 H 108 F 110 G 118 A 136

Problem D: Tsevie (Orange balloon)

The Tsevie village in Republic of Togo, is on fire due to the attack of the virtual enemy. Several places are already on fire and the fire is spreading fast to other places. Kossi who is the only person remaining alive in the war with the virtual enemy, tries to rescue himself by reaching to the only helicopter in the Tsevie village.

Suppose the Tsevie village is represented by an $n * m$ grid. At the initial time, some grid cells are on fire. If a cell catches fire at time x, all its 8 vertex-neighboring cells will catch fire at time $x + k$. If a cell catches fire, it will be on fire forever. At the initial time, Kossi stands at cell s and the helicopter is located at cell t. At any time x, Kossi can move from its current cell to one of four edge-neighboring cells, located at the left, right, top, or bottom of its current cell if that cell is not on fire at time $x + 1$. Note that each move takes one second.

Your task is to write a program to find the shortest path from s to t avoiding fire.

Standard Input

There are multiple test cases in the input. The first line of each test case contains three positive integers n, m and k ($1 \leqslant n, m, k \leqslant 100$), where n and m indicate the size of the test case grid n m, and k denotes the growth rate of fire. The next n lines, each contains a string of length m, where the jth character of the ith line represents the cell (i,j) of the grid. Cells which are on fire at time 0, are presented by character "`f`". There may exist no "`f`" in the test case. The helicopter and Kossi are located at cells presented by "`t`" and "`s`", respectively. Other cells are filled by "`-`" characters. The input terminates with a line containing "`0 0 0`" which should not be processed.

Standard Output

For each test case, output a line containing the shortest time to reach t from s avoiding fire. If it is impossible to reach t from s, write "`Impossible`" in the output.

Sample Input	Sample Output
7 7 2 f------ -f---f- ----f-- ------- ------f ---s--- t----f- 3 4 1 t--f --s- ---- 2 2 1 st f- 2 2 2 st f- 0 0 0	4 Impossible Impossible 1

Problem E : Addition

(purple balloon)

A multi-digit column addition is a formula on adding two integers written like this:

$$\begin{array}{r} 123 \\ +\ 456 \\ \hline 579 \end{array}$$

A multi-digit column addition is written on the blackboard, but the sum is not necessarily correct. We can erase any number of the columns so that the addition becomes correct. For example, in the following addition, we can obtain a correct addition by erasing the second and the forth columns.

$$\begin{array}{r} 12127 \\ +\ 45618 \\ \hline 51825 \end{array} \quad \Rightarrow \quad \begin{array}{r} 117 \\ +\ 468 \\ \hline 585 \end{array}$$

Your task is to find the minimum number of columns needed to be erased such that the remaining formula becomes a correct addition.

Standard Input

There are multiple test cases in the input. Each test case starts with a line containing the single integer n, the number of digit columns in the addition ($1 \leqslant n \leqslant 1000$). Each of the next 3 lines contain a string of n digits. The number on the third line is presenting the (not necessarily correct) sum of the numbers in the first and the second line. The input terminates with a line containing "0" which should not be processed.

Standard Output

For each test case, print a single line containing the minimum number of columns needed to be erased.

Sample Input	Sample Output
3 123 456 579 5 12127 45618 51825 2 24 32 32 5 12299 12299 25598 0	0 2 2 1