Problem Set for the 2nd Day of Summer Camp 2008

Japanese Alumni Group

Contest Held on 13 Sep 2008

Status of Problems

All problems were newly created by the members of Japanese Alumni Group.

Some portion of the problem statement may modified from the actual practice contest for correction and/or clarification.

Terms of Use

Problem A
Pi is Three

Input: A.txt

Summer Camp (Day 2), Tokyo
13 Sep 2008

π (spelled *pi* in English) is a mathematical constant representing the circumference of a circle whose diameter is one unit length. The name π is said to come from the first letter of the Greek words περιφέρεια (meaning periphery) and περίμετρος (perimeter).

Recently, the government of some country decided to allow use of 3, rather than 3.14, as the approximate value of π in school (although the decision was eventually withdrawn probably due to the blame of many people). This decision is very surprising, since this approximation is far less accurate than those obtained before the common era.

Ancient mathematicians tried to approximate the value of π without calculators. A typical method was to calculate the perimeter of inscribed and circumscribed regular polygons of the circle. For example, Archimedes (287–212 B.C.) proved that $223/71 < \pi < 22/7$ using 96-sided polygons, where $223/71$ and $22/7$ were both accurate to two fractional digits (3.14). The resultant approximation would be more accurate as the number of vertices of the regular polygons increased.

As you see in the previous paragraph, π was approximated by fractions rather than decimal numbers in the older ages. In this problem, you are requested to represent π as a fraction with the smallest possible denominator such that the representing value is not different by more than the given allowed error. If more than one fraction meets this criterion, the fraction giving better approximation is preferred.

▶ Input

The input consists of multiple datasets. Each dataset has a real number R ($0 < R \leq 1$) representing the allowed difference between the fraction and π. The value may have up to seven digits after the decimal point. The input is terminated by a line containing 0.0, which must not be processed.

▶ Output

For each dataset, output the fraction which meets the criteria in a line. The numerator and denominator of the fraction should be separated by a slash as shown in the sample output, and those numbers must be integers.

▶ Sample Input

```
0.15
0.05
0.0
```

▶ Output for the Sample Input

```
3/1
19/6
```

Problem B
Left Hand Rule

Input: B.txt

Summer Camp (Day 2), Tokyo
13 Sep 2008

The left-hand rule, which is also known as the wall follower, is a well-known strategy that solves a two-dimensional maze. The strategy can be stated as follows: once you have entered the maze, walk around with keeping your left hand in contact with the wall until you reach the goal. In fact, it is proven that this strategy solves some kind of mazes.

Your task is to write a program that determines whether the given maze is solvable by using the left-hand rule and (if the maze is solvable) the number of steps to reach the exit. Moving to a cell from the entrance or the adjacent (north, south, east or west) cell is counted as a step.

In this problem, the maze is represented by a collection of walls placed on the two-dimensional grid. We use an ordinary Cartesian coordinate system; the positive x-axis points right and the positive y-axis points up. Each wall is represented by a line segment which is parallel to the x-axis or the y-axis, such that both ends of each wall are located on integer coordinates. The size of the maze is given by W and H that indicate the width and the height of the maze, respectively. A rectangle whose vertices are on $(0,0)$, $(W,0)$, (W,H) and $(0,H)$ forms the outside boundary of the maze. The outside of the maze is always surrounded by walls except for the entrance of the maze. The entrance is represented by a line segment whose ends are (x_E, y_E) and (x_E', y_E'). The entrance has a unit length and is located somewhere on one edge of the boundary. The exit is a unit square whose bottom left corner is located on (x_X, y_X).

A few examples of mazes are illustrated in the figure below. They correspond to the datasets in the sample input.

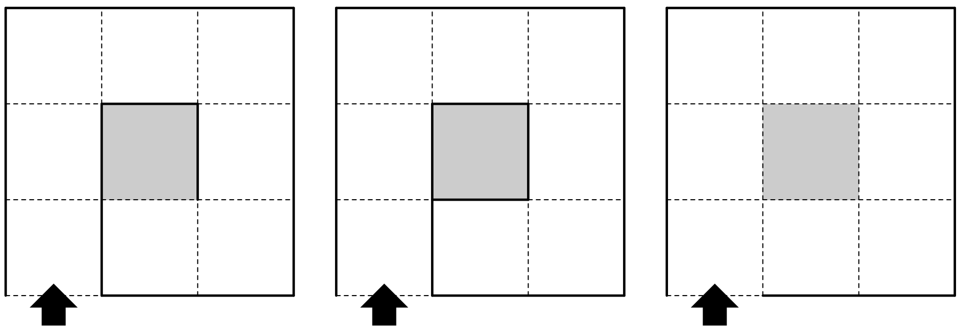

Figure 1: Example Mazes (shaded squares indicate the exits)

► Input

The input consists of multiple datasets.

Each dataset is formatted as follows:

W H N
x_1 y_1 x_1' y_1'
x_2 y_2 x_2' y_2'
...
x_N y_N x_N' y_N'
x_E y_E x_E' y_E' x_X y_X

W and H ($0 < W, H \leq 100$) indicate the size of the maze. N is the number of walls inside the maze. The next N lines give the positions of the walls, where (x_i, y_i) and (x_i', y_i') denote two ends of each wall ($1 \leq i \leq N$). The last line gives the positions of the entrance and the exit. You can assume that all the coordinates given in the input are integer coordinates and located inside the boundary of the maze. You can also assume that the wall description is not redundant, i.e. an endpoint of a wall is not shared by any other wall that is parallel to it.

The input is terminated by a line with three zeros.

► Output

For each dataset, print a line that contains the number of steps required to reach the exit. If the given maze is unsolvable, print "`Impossible`" instead of the number of steps.

► Sample Input

```
3 3 3
1 0 1 2
1 2 2 2
2 2 2 1
0 0 1 0 1 1
3 3 4
1 0 1 2
1 2 2 2
2 2 2 1
2 1 1 1
0 0 1 0 1 1
3 3 0
0 0 1 0 1 1
0 0 0
```

► Output for the Sample Input

```
9
Impossible
Impossible
```

Problem C
Alice and Bob

Input: C.txt

Summer Camp (Day 2), Tokyo
13 Sep 2008

Alice and Bob are in love with each other, but they have difficulty going out on a date — Alice is a very busy graduate student at the ACM university.

For this reason, Bob came to the ACM university to meet her on a day. He tried to reach the meeting spot but got lost because the campus of the university was very large. Alice talked with him via mobile phones and identified his current location exactly. So she told him to stay there and decided to go to the place where she would be visible to him without interruption by buildings.

The campus can be considered as a two-dimensional plane and all buildings as rectangles whose edges are parallel to x-axis or y-axis. Alice and Bob can be considered as points. Alice is visible to Bob if the line segment connecting them does not intersect the interior of any building. Note that she is still visible even if the line segment touches the borders of buildings.

Since Alice does not like to walk, she wants to minimize her walking distance. Can you write a program that finds the best route for her?

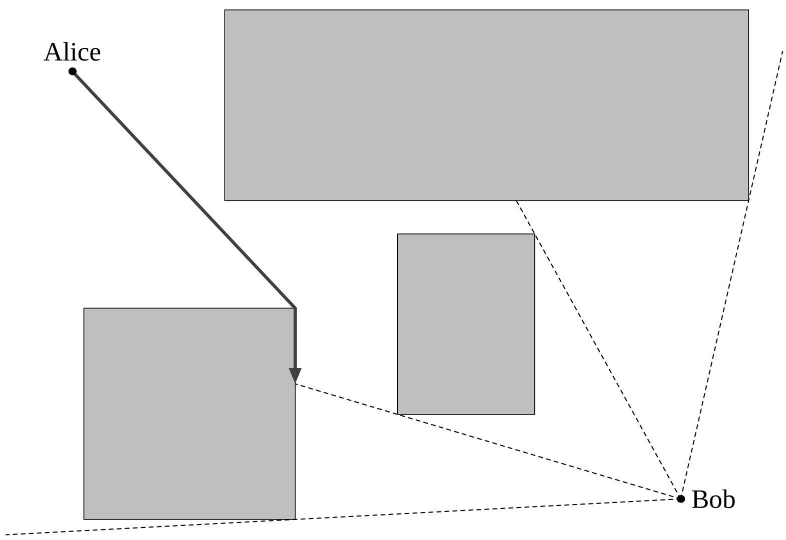

Figure 2: Example Situation

► Input

The input contains multiple datasets. The end of the input is indicated by a line containing a single zero. Each dataset is formatted as follows.

N
x_{11} y_{11} x_{12} y_{12}
x_{21} y_{21} x_{22} y_{22}
$\dots$
x_{N1} y_{N1} x_{N2} y_{N2}
Ax Ay Bx By

N ($0 < N \leq 30$) is the number of buildings. The i-th building is given by its bottom left corner (x_{i1}, y_{i1}) and up right corner (x_{i2}, y_{i2}). (Ax, Ay) is the location of Alice and (Bx, By) is that of Bob. All integers x_{i1}, y_{i1}, x_{i2}, y_{i2}, Ax, Ay, Bx and By are between -1000 and 1000, inclusive. You may assume that no building touches or overlaps other buildings.

► Output

For each dataset, output a separate line containing the minimum distance Alice has to walk.

The value may contain an error less than or equal to 0.001. You may print any number of digits after the decimal point.

► Sample Input

```
1
3 3 7 7
2 2 8 2
2
2 5 5 9
6 1 9 5
1 5 10 5
2
2 1 3 2
2 3 3 4
1 1 4 4
1
3 3 7 7
1 5 9 5
1
3 3 7 7
1 5 8 5
1
3 3 7 7
1 5 10 5
0
```

► Output for the Sample Input

```
0.000
0.000
0.000
5.657
6.406
4.992
```

Problem D
Deadly Dice Game

Input: D.txt

Summer Camp (Day 2), Tokyo
13 Sep 2008

T.I. Financial Group, a world-famous group of finance companies, has decided to hold an evil gambling game in which insolvent debtors compete for special treatment of exemption from their debts.

In this game, each debtor starts from one cell on the stage called the Deadly Ring. The Deadly Ring consists of N cells and each cell is colored black or red. Each cell is connected to exactly two other adjacent cells and all the cells form a ring. At the start of the game, each debtor chooses which cell to start. Then he rolls a die and moves on the ring in clockwise order by cells as many as the number of spots shown on the upside of the die. This step is called a round, and each debtor repeats a round T times. A debtor will be exempted from his debt if he is standing on a red cell after he finishes all the rounds. On the other hand, if he finishes the game at a black cell, he will be sent somewhere else and forced to devote his entire life to hard labor.

You have happened to take part in this game. As you are allowed to choose the starting cell, you want to start from a cell that maximizes the probability of finishing the game at a red cell. Fortunately you can bring a laptop PC to the game stage, so you have decided to write a program that calculates the maximum winning probability.

► Input

The input consists of multiple datasets.

Each dataset consists of two lines. The first line contains two integers N ($1 \leq N \leq 2000$) and T ($1 \leq T \leq 2000$) in this order, delimited with a single space. The second line contains a string of N characters that consists of characters '`R`' and '`B`', which indicate red and black respectively. This string represents the colors of the cells in clockwise order.

The input is terminated by a line with two zeros. This is not part of any datasets and should not be processed.

► Output

For each dataset, print the maximum probability of finishing the game at a red cell in one line. Your program may output an arbitrary number of digits after the decimal point, provided that the absolute error does not exceed 10^{-8}.

► Sample Input

```
6 1
RBRBRB
10 1
RBBBRBBBBB
10 2
RBBBRBBBBB
10 10
RBBBBBBBBB
0 0
```

► Output for the Sample Input

```
0.50000000
0.33333333
0.22222222
0.10025221
```

Problem E
Reverse a Road

Input: E.txt

Summer Camp (Day 2), Tokyo
13 Sep 2008

Andrew R. Klein resides in the city of Yanwoe, and goes to his working place in this city every weekday. He has been totally annoyed with the road traffic of this city. All the roads in this city are one-way, so he has to drive a longer way than he thinks he need.

One day, the following thought has come up to Andrew's mind: "How about making the sign of one road indicate the opposite direction? I think my act won't be out as long as I change just one sign. Well, of course I want to make my route to the working place shorter as much as possible. Which road should I alter the direction of?" What a clever guy he is.

You are asked by Andrew to write a program that finds the shortest route when the direction of up to one road is allowed to be altered. You don't have to worry about the penalty for complicity, because you resides in a different country from Andrew and cannot be punished by the law of his country. So just help him!

► Input

The input consists of a series of datasets, each of which is formatted as follows:

N
S T
M
A_1 B_1
A_2 B_2
...
A_M B_M

N denotes the number of points. S and T indicate the points where Andrew's home and working place are located respectively. M denotes the number of roads. Finally, A_i and B_i indicate the starting and ending points of the i-th road respectively. Each point is identified by a unique number from 1 to N. Some roads may start and end at the same point. Also, there may be more than one road connecting the same pair of starting and ending points.

You may assume all the following: $1 \leq N \leq 1000$, $1 \leq M \leq 10000$, and $S \neq T$.

The input is terminated by a line that contains a single zero. This is not part of any dataset, and hence should not be processed.

► Output

For each dataset, print a line that contains the shortest distance (counted by the number of passed roads) and the road number whose direction should be altered. If there are multiple ways to obtain the shortest distance, choose one with the smallest road number. If no direction change results in a shorter route, print 0 as the road number.

Separate the distance and the road number by a single space. No extra characters are allowed.

► Sample Input

```
4
1 4
4
1 2
2 3
3 4
4 1
0
```

► Output for the Sample Input

```
1 4
```

Problem F
Webby Subway

Input: F.txt

Summer Camp (Day 2), Tokyo
13 Sep 2008

You are an officer of the Department of Land and Transport in Oykot City. The department has a plan to build a subway network in the city central of Oykot.

In the plan, n subway lines are built, and each line has two or more stations. Because of technical problems, a rail track between two stations should be straight and should not have any slope. To make things worse, no rail track can contact with another rail track, even on a station. In other words, two subways on the same floor cannot have any intersection.

Your job is to calculate the least number of required floors in their plan.

► Input

The input consists of multiple datasets. Each dataset is formatted as follows:

N
$Line_1$
$Line_2$
$\ldots$
$Line_N$

Here, N is a positive integer that indicates the number of subway lines to be built in the plan ($N \leq 22$), and $Line_i$ is the description of the i-th subway line with the following format:

S
X_1 Y_1
X_2 Y_2
$\ldots$
X_S Y_S

S is a positive integer that indicates the number of stations in the line ($S \leq 30$), and (X_i, Y_i) indicates the coordinates of the i-th station of the line ($-10000 \leq X_i, Y_i \leq 10000$). The rail tracks are going to be built between two consecutive stations in the description. No stations of the same line have the same coordinates.

The input is terminated by a dataset of $N = 0$, and it should not be processed.

► Output

For each dataset, you should output the least number of required floors.

► Sample Input

```
2
2
0 0
10 0
2
0 10
10 10
2
2
0 0
10 10
2
0 10
10 0
3
2
0 0
10 10
2
0 10
10 0
2
1 0
1 10
0
```

► Output for the Sample Input

```
1
2
3
```

Problem G
Time Trial

Input: G.txt

Summer Camp (Day 2), Tokyo
13 Sep 2008

Some people like finishing computer games in an extremely short time. Terry A. Smith is one of such and prefers role playing games particularly.

He is now trying to find a shorter play for one of the key events in a role playing game. In this event, a player is presented a kind of puzzle on a grid map with three rocks and three marked squares. The objective is to have all the rocks placed on the marked squares by controlling the hero of this game appropriately.

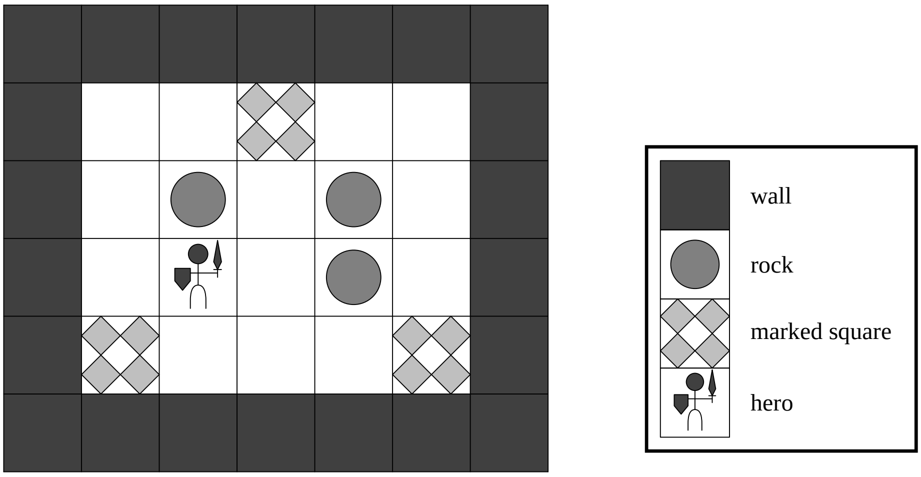

Figure 3: Example Map

The hero can move to the four adjacent squares, that is, to the north, east, south, and west, unless his move is blocked by walls or rocks. He can never enter squares occupied by walls. On the other hand, when he is moving to a square occupied by a rock, he pushes the rock in his moving direction. Nonetheless, he cannot push the rock if the next square is occupied by a wall or another rock and his move is blocked in this case. Also, he can only move one rock at a time. It is allowed to have rocks pass through marked squares.

Terry thinks he can reduce his playing time by finding the optimal way to move the rocks and then playing the event accordingly. However, it is too hard for him to find the solution of this puzzle by hand. So you are asked by him to write a program that finds the smallest number of steps for the maps given as the input. Here, each move from a square to its adjacent square is counted as one step.

► Input

The input is a sequence of datasets. Each dataset has the following format:

> W H
> Row_1
> ...
> Row_H

W and H are the width and height of the map ($4 \leq W, H \leq 16$). Row_i denotes the i-th row of the map and consists of W characters. Each character represents a square and is one of the following: '`#`' (wall),

'.' (floor), '*' (rock), '_' (marked square), and '@' (hero). Each map contains exactly three rocks, three marked squares, and one hero. The outermost squares are always occupied by walls. You may assume that the number of non-wall squares does not exceed fifty. It is also guaranteed that there is at least one solution for every map.

The input is terminated by a line with two zeros. This line is not part of any datasets and should not be processed.

► Output

For each dataset, print the smallest number of steps in a line.

► Sample Input

```
7 6
#######
#.._..#
#.*.*.#
#.@.*.#
#_..._#
#######
10 13
##########
####___###
####...###
####...###
#####.####
#.....#..#
#.#*.*.*.#
#...###..#
###.#.#.##
###.#.#.##
###.....##
###..@..##
##########
0 0
```

► Output for the Sample Input

```
15
118
```

Problem H
Vending Machine

Input: H.txt

Summer Camp (Day 2), Tokyo
13 Sep 2008

There has been marketing warfare among beverage vendors, and they have been working hard for increase of their sales. The Kola-Coqua Company is one of the most successful vendors among those: their impressive advertisements toward the world has brought the overwhelming market share of their representative product called Koque.

This time, Kola-Coqua is focusing on vending machines. They think cusomters will be more pleasant as the machines respond more quickly, so they have improved many parts of the machines.

In particular, they have developed a new device of change return. The new device can give one or more kinds of coins at a time (in a single operation), although it can give only one coin for each kind at once. For example, suppose there are 500-yen, 100-yen, 50-yen and 10-yen coins, change of 6540 yen can be made by four operations of giving 500-yen and 10-yen coins and nine operations of giving 500-yen coins. In conclusion, 6540 yen can be returned by thirteen operations. It is supposed that the new device allows customers to make their purchase more quickly and so helps Kola-Coqua's market share grow up.

However, the project leader says "No, it's not optimal yet." His suggesion is as follows: the real optimization is to minimize the number of operations. For example, change of 6540 yen should be made with ten of 500-yen coins, ten of 100-yen coins, ten of 50-yen coins, and four of 10-yen coins. This way, 6540 yen can be returned only with ten operations. This allows full speed-up in giving back change, even though it sometimes results in a huge amount of coins.

Given which kinds of coins are available and how much change should be given back, you are to write a program that calculates the minimum number of operations according to the above suggestion. You may assume that there are enough amount of coins inside the vending machines.

► Input

The input consists of multiple data sets. Each dataset is described by two lines. The first line contains N ($N \leq 10$) and M ($M \leq 100000$) indicating the number of kinds of coins and the amount of change to be made, respectively. The second line contains N integers representing the value of each kind of coin.

The input is terminated by a dataset of $N = M = 0$. This dataset must not be processed.

► Output

For each dataset, output in a line the minimum number of operations needed to give back exactly the specified amount of change.

► Sample Input

```
6 330
1 5 10 50 100 500
7 127
1 2 4 8 16 32 64
2 10000
1000 2000
0 0
```

► Output for the Sample Input

```
2
1
4
```

Problem I
Memory Match

Input: I.txt

Summer Camp (Day 2), Tokyo
13 Sep 2008

Memory match is a single-player game which employs a set of $2M$ cards. Each card is labeled with a number between 1 and M on its face. For each number i ($1 \le i \le M$), there are exactly two cards which have the number i. At the start of the game, all cards are shuffled and laid face down on a table. In each turn you choose two cards and turn them face up. If two numbers on the cards are the same, they are removed from the table. Otherwise, they are turned face down again (this is called a mismatch). When you choose cards, you do not have to turn two cards simultaneously; you can choose the second card after you see the number of the first card. The objective of the game is to remove all the cards with as few mismatches as possible.

Royce A. Mitchell has extraordinary memory, so he can remember all the positions and the numbers of the cards that he has already turned face up. Your task is to write a program that calculates the expected number of mismatches, on average, when he plays the game optimally.

► Input

The input consists of multiple datasets.

Each dataset consists of one even number N ($2 \le N \le 1000$) which denotes the number of cards in the set.

The end of input is indicated by a line that contains a single zero. This is not part of the input and you may not treat this line as a dataset.

► Output

For each dataset, print the expected number of mismatches. Each output value may have an arbitrary number of fractional digits, provided that the error is within 10^{-6}.

► Sample Input

```
2
4
6
8
10
52
0
```

► Output for the Sample Input

```
0.0000000000
0.6666666667
1.3333333333
1.9238095238
2.5523809524
15.4435236099
```

Problem J
Tile Puzzle

Input: J.txt

Summer Camp (Day 2), Tokyo
13 Sep 2008

You are visiting Ancient and Contemporary Museum. Today there is held an exhibition on the history of natural science. You have seen many interesting exhibits about ancient, medieval, and modern science and mathematics, and you are in a resting space now.

You have found a number of panels there. Each of them is equipped with $N \times N$ electric tiles arranged in a square grid. Each tile is lit in one of the following colors: black (unlit), red, green, yellow, blue, magenta, and cyan. Initially all the tiles are in black. When a tile is touched on, that tile and the eight adjacent tiles will change their colors as follows: black → red, red → green, green → yellow, yellow → blue, blue → magenta, magenta → cyan, and cyan → black. Here, the leftmost and rightmost columns are considered adjacent, and so as the uppermost and lowermost rows. There is a goal pattern for each panel, and you are to change the colors of the tiles as presented in the goal pattern. For example, if you are given the goal pattern shown in the figure below for a panel of 4×4, you will touch on the upper-left tile once and then on the lower-right tile twice (note that this might not be the only way).

Since you are good at programming, you guess you can find the solution using your computer. So your job in this problem is to write a program for it.

yellow	green	red	yellow
green	green	black	green
red	black	red	red
yellow	green	red	yellow

Figure 4: Example Goal Pattern

▶ Input

The input contains a series of datasets. Each dataset is given in the following format:

N
Row_1
...
Row_N

N indicates the size (i.e. the number of rows and columns) of the electrical panel ($3 \leq N \leq 15$). Row_i describes the goal pattern of the i-th row and contains exactly N numbers separated by a space. The j-th number indicates the color of the j-th column, and it is one of the following: 0 (denoting black), 1 (red), 2 (green), 3 (yellow), 4 (blue), 5 (magenta), and 6 (cyan).

The input is terminated by a line containing a single zero. This line is not part of any datasets.

► Output

For each dataset, your program should produce the output of N lines. The i-th line should correspond to the i-th row and contain exactly N numbers separated by a space, where the j-th number should be the number of touches on the tile of the j-th column. The number should be in the range from 0 to 6 inclusive.

If there is more than one solution, your program may output any of them. If it is impossible to make the goal pattern, your program should output a single line containing "`-1`" (without quotes) instead of the N lines.

A blank line should follow the output for every dataset (including the last one).

► Sample Input

```
4
3 1 2 3
1 1 0 1
2 0 2 2
3 1 2 3
5
3 3 3 0 0
3 3 3 0 0
3 3 0 4 4
0 0 4 4 4
0 0 4 4 4
0
```

► Output for the Sample Input

```
1 0 0 0
0 0 0 0
0 0 0 0
0 0 0 2

0 0 0 0 0
0 3 0 0 0
0 0 0 0 0
0 0 0 4 0
0 0 0 0 0

```