The 23rd Annual

ACM International Collegiate Programming Contest WORLD FINALS

Problem A

Bee Breeding

Input: bees.in

Professor B. Heif is conducting experiments with a species of South American bees that he found during an expedition to the Brazilian rain forest. The honey produced by these bees is of superior quality compared to the honey from European and North American honey bees. Unfortunately, the bees do not breed well in captivity. Professor Heif thinks the reason is that the placement of the different maggots (for workers, queens, etc.) within the honeycomb depends on environmental conditions, which are different in his laboratory and the rain forest.

As a first step to validate his theory, Professor Heif wants to quantify the difference in maggot placement. For this he measures the distance between the cells of the comb into which the maggots are placed. To this end, the professor has labeled the cells by marking an arbitrary cell as number 1, and then labeling the remaining cells in a clockwise fashion, as shown in the following figure.

```
       __    __    __    __
    __/  \__/  \__/  \__/  \__
 __/  \__/  \__/53\__/  \__/  \__
/  \__/  \__/52\__/54\__/  \__/  \
\__/  \__/51\__/31\__/55\__/  \__/
/  \__/50\__/30\__/32\__/56\__/  \
\__/49\__/29\__/15\__/33\__/57\__/
/  \__/28\__/14\__/16\__/34\__/  \
\__/48\__/13\__/ 5\__/17\__/58\__/
/..\__/27\__/ 4\__/ 6\__/35\__/  \
\__/47\__/12\__/ 1\__/18\__/59\__/
/..\__/26\__/ 3\__/ 7\__/36\__/  \
\__/46\__/11\__/ 2\__/19\__/60\__/
/..\__/25\__/10\__/ 8\__/37\__/  \
\__/45\__/24\__/ 9\__/20\__/61\__/
/..\__/44\__/23\__/21\__/38\__/  \
\__/70\__/43\__/22\__/39\__/62\__/
/  \__/69\__/42\__/40\__/63\__/  \
\__/  \__/68\__/41\__/64\__/  \__/
/  \__/  \__/67\__/65\__/  \__/  \
\__/  \__/  \__/66\__/  \__/  \__/
   \__/  \__/  \__/  \__/  \__/
      \__/  \__/  \__/  \__/
```

For example, two maggots in cells 19 and 30 are 5 cells apart. One of the shortest paths connecting the two cells is via the cells 19 - 7 - 6 - 5 - 15 - 30, so you must move five times to adjacent cells to get from 19 to 30.

Professor Heif needs your help to write a program that computes the distance, defined as the number of cells in a shortest path, between any pair of cells.

Input

The input consists of several lines, each containing two integers a and b ($a,b \leq 10000$), denoting numbers of cells. The integers are always positive, except in the last line where $a=b=0$ holds. This last line terminates the input and should not be processed.

Output

For each pair of numbers (*a*,*b*) in the input file, output the distance between the cells labeled *a* and *b*. The distance is the minimum number of moves to get from *a* to *b*.

Sample Input	Output for the Sample Input
`19 30` `0 0`	`The distance between cells 19 and 30 is 5.`

The 23rd Annual

ACM International Collegiate Programming Contest WORLD FINALS

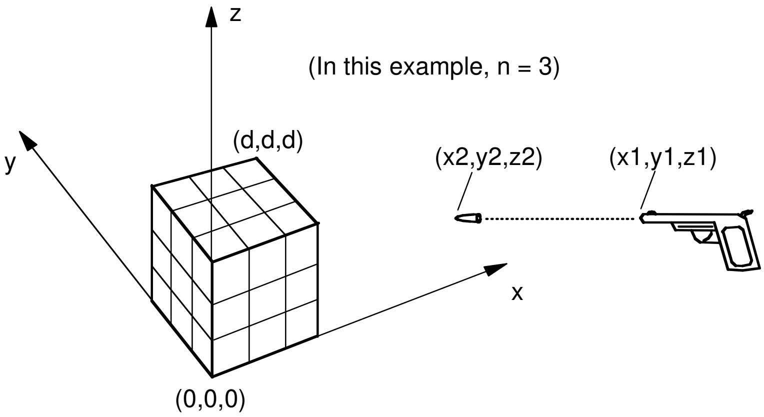
acm International Collegiate Programming Contest

IBM event sponsor

Problem B

Bullet Hole

Input: bullet.in



A cube is suspended in space. A Cartesian coordinate system is defined with its origin at one of the bottom corners of the cube, as shown in the figure. The cube has side dimension d, so its opposite corners are at coordinates (0, 0, 0) and (d, d, d). The positive z-direction of the coordinate system is "up" with respect to gravity.

The interior of the cube contains partitions with uniform spacing in each dimension, so that the cube is partitioned into n^3 mini-cubes of equal size. The partitions are thin and watertight, and each mini-cube is filled with water. The total volume of water in all the minicubes is d^3.

A gun fires a bullet which may hit the cube. The muzzle of the gun is at the point (x_1, y_1, z_1). The point (x_2, y_2, z_2) is a point on the bullet's path that defines the direction of the bullet. The bullet does not shatter the cube, but wherever the bullet touches a side or interior partition of the cube, it makes a small hole. Bullet holes may be made in the sides, edges, or corners of the interior mini-cubes. Water, influenced by gravity, may leak through these small holes. All the water that leaks out of the large cube is collected and measured.

Input

The input data set consists of several trials. Each trial is described by eight integers. The first integer is n ($n \le 50$), as described above. The second integer is d ($d \le 100$). The remaining six integers—x_1, y_1, z_1, x_2, y_2, z_2—represent the origin and a point on the path of the bullet ($-100 \le x_1, y_1, z_1, x_2, y_2, z_2 \le 100$). The origin and the point on the path of the bullet are not the same. After the last trial, the integer 0 terminates the data set.

Output

Your program must compute the total volume of water that leaks out of the large cube. For each trial, print the trial number, the notation `Volume =`, and the total volume of water accurate to two digits to the right of the decimal point. Print a blank line between trials.

Note

In this problem, two real numbers are considered equal if they are less than 10^{-6} apart.

Sample Input	Output for the Sample Input

` 5   25     5  15   0     5  15 100`
` 3   30     0 -35   0     3 -25   3`
`10   16     8  17  11    12  19   6`
` 0`
	`Trial 1, Volume = 2500.00`
	
	`Trial 2, Volume = 1995.00`
	
	`Trial 3, Volume = 0.00`

The 23rd Annual

ACM International Collegiate Programming Contest WORLD FINALS

Problem C

A Dicey Problem

Input: dicemaze.in

The three-by-three array in Figure 1 is a maze. A standard six-sided die is needed to traverse the maze (the layout of a standard six-sided die is shown in Figure 2). Each maze has an initial position and an initial die configuration. In Figure 1, the starting position is row 1, column 2—the "2" in the top row of the maze—and the initial die configuration has the "5" on top of the die and the "1" facing the player (assume the player is viewing the maze from the bottom edge of the figure).

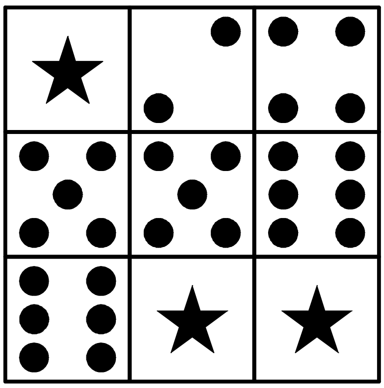


Figure 1: Sample Dice Maze

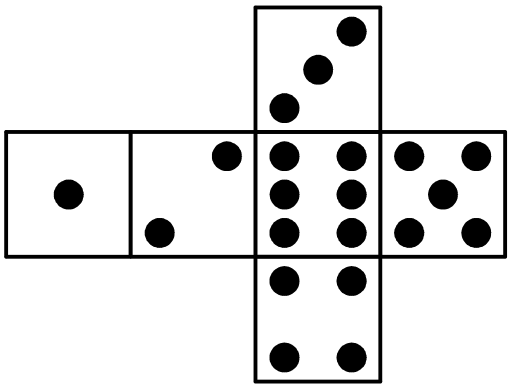


Figure 2: Standard Layout of Six-Sided Die

To move through the maze you must tip the die over on an edge to land on an adjacent square, effecting horizontal or vertical movement from one square to another. However, you can only move onto a square that contains the same number as the number displayed on the top of the die before the move, or onto a "wild" square which contains a star. Movement onto a wild square is always allowed regardless of the number currently displayed on the top of the die. The goal of the maze is to move the die off the starting square and to then find a way back to that same square.

For example, at the beginning of the maze there are two possible moves. Since the 5 is on top of the die, it is possible to move down one square, and since the square to the left of the starting position is wild it is also possible to move left. If the first move chosen is to move down, this brings the 6 to the top of the die and moves are now possible both to the right and down. If the first move chosen is instead to the left, this brings the 3 to the top of the die and no further moves are possible.

If we consider maze locations as ordered pairs of row and column numbers (*row*, *column*) with row indexes starting at 1 for the top row and increasing toward the bottom, and column indexes starting at 1 for the left column and increasing to the right, the solution to this simple example maze can be specified as: (1,2), (2,2), (2,3), (3,3), (3,2), (3,1), (2,1), (1,1), (1,2). A bit more challenging example maze is shown in Figure 3.

The goal of this problem is to write a program to solve dice mazes. The input file will contain several mazes for which the program should search for solutions. Each maze will have either a unique solution or no solution at all. That is, each maze in the input may or may not have a solution, but those with a solution are guaranteed to have only one unique solution. For each input maze, either a solution or a message indicating no solution is possible will be sent to the output.

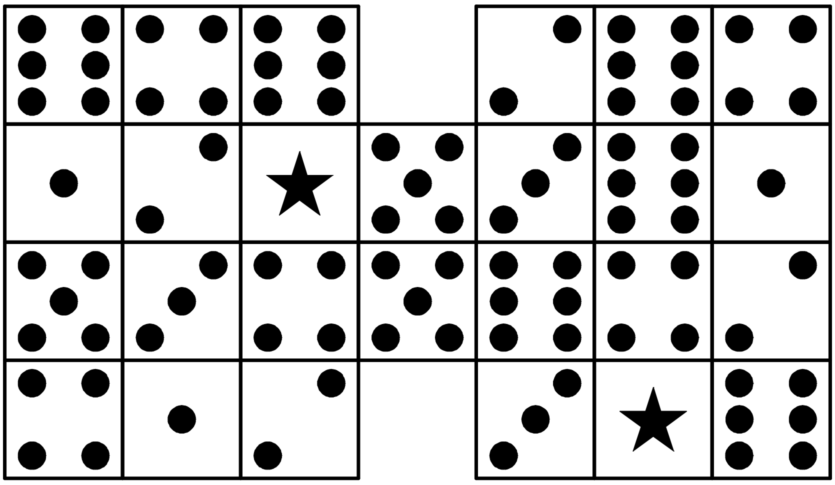


Figure 3: Start at (2,6) with the 3 on top and the 6 facing you.

Input

The input file begins with a line containing a string of no more than 20 non-blank characters that names the first maze. The next line contains six integers delimited by single spaces. These integers are, in order, the number of rows in the maze (an integer from 1 to 10, call this value R), the number of columns in the maze (an integer from 1 to 10, call this value C), the starting row, the starting column, the number that should be on top of the die at the starting position, and finally the number that should be facing you on the die at the starting position. The next R lines contain C integers each, again delimited by single spaces. This $R \times C$ array of integers defines the maze. A value of zero indicates an empty location in the maze (such as the two empty squares in the center column of the maze in Figure 3), and a value of -1 indicates a wild square. This input sequence is repeated for each maze in the input. An input line containing only the word "`END`" (without the quotes) as the name of the maze marks the end of the input.

Output

The output should contain the name of each maze followed by its solution or the string "No Solution Possible" (without the quotes). All lines in the output file except for the maze names should be indented exactly two spaces. Maze names should start in the leftmost column. Solutions should be output as a comma-delimited sequence of the consecutive positions traversed in the solution, starting and ending with the same square (the starting square as specified in the input). Positions should be specified as ordered pairs enclosed in parentheses. The solution should list 9 positions per line (with the exception of the last line of the solution for which there may not be a full 9 positions to list), and no spaces should be present within or between positions.

Sample Input

```
DICEMAZE1
3 3 1 2 5 1
-1 2 4
5 5 6
6 -1 -1
DICEMAZE2
4 7 2 6 3 6
6 4 6 0 2 6 4
1 2 -1 5 3 6 1
5 3 4 5 6 4 2
4 1 2 0 3 -1 6
DICEMAZE3
3 3 1 1 2 4
2 2 3
4 5 6
-1 -1 -1
END
```

Output for the Sample Input

```
DICEMAZE1
  (1,2),(2,2),(2,3),(3,3),(3,2),(3,1),(2,1),(1,1),(1,2)
DICEMAZE2
  (2,6),(2,5),(2,4),(2,3),(2,2),(3,2),(4,2),(4,1),(3,1),
  (2,1),(2,2),(2,3),(2,4),(2,5),(1,5),(1,6),(1,7),(2,7),
  (3,7),(4,7),(4,6),(3,6),(2,6)
DICEMAZE3
  No Solution Possible
```

The 23rd Annual
ACM International Collegiate Programming Contest WORLD FINALS

Problem D
The Fortified Forest
Input: forest.in

Once upon a time, in a faraway land, there lived a king. This king owned a small collection of rare and valuable trees, which had been gathered by his ancestors on their travels. To protect his trees from thieves, the king ordered that a high fence be built around them. His wizard was put in charge of the operation.

Alas, the wizard quickly noticed that the only suitable material available to build the fence was the wood from the trees themselves. In other words, it was necessary to cut down some trees in order to build a fence around the remaining trees. Of course, to prevent his head from being chopped off, the wizard wanted to minimize the value of the trees that had to be cut. The wizard went to his tower and stayed there until he had found the best possible solution to the problem. The fence was then built and everyone lived happily ever after.

You are to write a program that solves the problem the wizard faced.

Input

The input contains several test cases, each of which describes a hypothetical forest. Each test case begins with a line containing a single integer n, $2 \le n \le 15$, the number of trees in the forest. The trees are identified by consecutive integers 1 to n. Each of the subsequent n lines contains 4 integers x_i, y_i, v_i, l_i that describe a single tree. (x_i, y_i) is the position of the tree in the plane, v_i is its value, and l_i is the length of fence that can be built using the wood of the tree. v_i and l_i are between 0 and 10,000.

The input ends with an empty test case ($n = 0$).

Output

For each test case, compute a subset of the trees such that, using the wood from that subset, the remaining trees can be enclosed in a single fence. Find the subset with minimum value. If more than one such minimum-value subset exists, choose one with the smallest number of trees. For simplicity, regard the trees as having zero diameter.

Display, as shown below, the test case numbers (1, 2, …), the identity of each tree to be cut, and the length of the excess fencing (accurate to two fractional digits).

Display a blank line between test cases.

Sample Input

```
6
 0  0  8  3
 1  4  3  2
 2  1  7  1
 4  1  2  3
 3  5  4  6
 2  3  9  8
3
 3  0 10  2
 5  5 20 25
 7 -3 30 32
0
```

Output for the Sample Input

```
Forest 1
Cut these trees: 2 4 5
Extra wood: 3.16

Forest 2
Cut these trees: 2
Extra wood: 15.00
```

The 23rd Annual
ACM International Collegiate
Programming Contest
WORLD FINALS

Problem E
Trade on Verweggistan
Input: prul.in

Since the days of Peter Stuyvesant and Abel Tasman, Dutch merchants have been traveling all over the world to buy and sell goods. Once there was some trade on Verweggistan, but it ended after a short time. After reading this story you will understand why.

At that time Verweggistan was quite popular, because it was the only place in the world where people knew how to make a ‘prul’. The end of the trade on Verweggistan meant the end of the trade in pruls (or ‘prullen’, as the Dutch plural said), and very few people nowadays know what a prul actually is.

Pruls were manufactured in workyards. Whenever a prul was finished it was packed in a box, which was then placed on top of the pile of previously produced pruls. On the side of each box the price was written. The price depended on the time it took to manufacture the prul. If all went well, a prul would cost one or two florins, but on a bad day the price could easily rise to 15 florins or more. This had nothing to do with quality; all pruls had the same value.

In those days pruls sold for 10 florins each in Holland. Transportation costs were negligible since the pruls were taken as extra on ships that would sail anyway. When a Dutch merchant went to Verweggistan, he had a clear purpose: buy pruls, sell them in Holland, and maximize his profits. Unfortunately, the Verweggistan way of trading pruls made this more complicated than one would think.

One would expect that merchants would simply buy the cheapest pruls, and the pruls that cost more than 10 florins would remain unsold. Unfortunately, all workyards on Verweggistan sold their pruls in a particular order. The box on top of the pile was sold first, then the second one from the top, and so on. So even if the fifth box from the top was the cheapest one, a merchant would have to buy the other four boxes above to obtain it.

As you can imagine, this made it quite difficult for the merchants to maximize their profits by buying the right set of pruls. Not having computers to help with optimization, they quickly lost interest in trading pruls at all.

In this problem, you are given the description of several workyard piles. You have to calculate the maximum profit a merchant can obtain by buying pruls from the piles according to the restrictions given above. In addition, you have to determine the number of pruls he has to buy to achieve this profit.

Input

The input describes several test cases. The first line of input for each test case contains a single integer w, the number of workyards in the test case ($1 \le w \le 50$).

This is followed by w lines, each describing a pile of pruls. The first number in each line is the number b of boxes in the pile ($0 \le b \le 20$). Following it are b positive integers, indicating the prices (in florins) of the pruls in the stack, given from top to bottom.

The input is terminated by a description starting with $w = 0$. This description should not be processed.

Output

For each test case, print the case number (1, 2, …). Then print two lines, the first containing the maximum profit the merchant can achieve. The second line should specify the number of pruls the merchant has to buy to obtain this profit. If this number is not uniquely determined, print the possible values in increasing order. If there are more than ten possible values, print only the 10 smallest.

Display a blank line between test cases.

Sample Input	Output for the Sample Input

<pre>1
6 12 3 10 7 16 5
2
5 7 3 11 9 10
9 1 2 3 4 10 16 10 4 16
0</pre>
	<pre>Workyards 1
	Maximum profit is 8.
	Number of pruls to buy: 4
	
	Workyards 2
	Maximum profit is 40.
	Number of pruls to buy: 6 7 8 9 10 12 13</pre>

The 23rd Annual
ACM International Collegiate Programming Contest WORLD FINALS

Problem F
Robot
Input: robot.in

A robot arm used in an automated factory consists of N connected links: $link_1$ which is connected to $link_2, \ldots,$ and $link_{N-1}$ which is connected to $link_N$. Each link is a straight rod of a specified length, len_1, len_2, …, len_N. Between each pair of connected links is a servo, $servo_2$ (between $link_1$ and $link_2$), …, and $servo_N$ (between $link_{N-1}$ and $link_N$) that can be activated to adjust the angle between the connected links. $Link_1$ is also connected by a servo, $servo_1$, to the factory floor (at the point x=0, y=0, z=0 in a Cartesian coordinate system). At the free (unconnected) end of the last link ($link_N$) is a "hand" that can be used to grasp objects.

In the initial setting of the robot arm, each servo is set to no rotation (0 degrees), and the links in the robot arm coincide with the z-axis. The xy plane is horizontal (the factory floor), and the entire robot arm is initially pointing up, vertically. From this initial setting, each servo can effect a rotation of up to 90 degrees in either of two directions. $Servo_1$ moves the entire robot arm in the xz plane by rotation about the y-axis. $Servo_2$ moves the arm (except $link_1$) in the (perhaps rotated) yz plane by rotation about the x-axis. In a similar manner, each odd-numbered servo can rotate the remaining part of the arm in the (perhaps rotated) xz plane, and each even-numbered servo can rotate the remaining part of the arm in the (perhaps rotated) yz plane. In effect, the servos rotate the links about the y and x-axes of coordinate systems fixed to the end of each link. Counterclockwise rotations about a coordinate axis are produced with positive rotation angles, if we are looking along the positive half of the axis toward the coordinate origin. The sample data has been carefully chosen to illustrate the effects of these rotations.

There are two restrictions on the final positioning of the robot's arm. No part of the arm can be below the factory floor, and the links in the robot's arm cannot intersect with each other (except where they are connected by the servos).

You should check only the final position of the arm.

Given the number of links in a robot's arm, their lengths, and the proposed settings of the servos, first determine if the proposed positioning of the arm is allowable. If the arm can be positioned as proposed, then determine the coordinates of the robot's hand, accurate to three fractional digits. Otherwise identify the first (smallest numbered) servo that has an inappropriate setting, and why that setting is inappropriate. Links are assumed to intersect if they come within 0.001 length units of each other.

Input
The input data will contain multiple test cases. Each test case includes, in order, the number of links, N, their lengths, len_1, …, len_N, and the proposed angles to which the servos (starting with $servo_1$) are to be set. The lengths and servo angles are real numbers, and the number of links is an integer. There will be no more than 10 links in any robot arm. The last test case is followed by a negative integer.

Output
For each test case, display the test case number (starting with 1). Then, if the proposed setting is allowable, display the position of the robot's hand in the original (factory floor) coordinate system (with three fractional digits). Otherwise display the identity of the first servo with an inappropriate setting and why that setting is inappropriate. An output format similar to that shown below is acceptable.

Sample Input	Output for the Sample Input
`2 25 15 0 90.0`	`Case 1: robot's hand is at (0.000,-15.000,25.000)`
`1 1.0 45.0`	`Case 2: robot's hand is at (0.707,0.000,0.707)`
`2 1 1 0 45`	`Case 3: robot's hand is at (0.000,-0.707,1.707)`
`4 1 2 3 4 90 0 0 0`	`Case 4: robot's hand is at (10.000,0.000,0.000)`
`3 1 1 1 0 90 90`	`Case 5: robot's hand is at (1.000,-1.000,1.000)`
`2 1 1 45.0 45`	`Case 6: robot's hand is at (1.207,-0.707,1.207)`
`4 1 1 1 2 0 90 0 90`	`Case 7: servo 4 attempts to move arm below floor`
`8 10 1 1 1 1 1 1 2`	`Case 8: servo 8 causes link collision`
`   0 0 90 0 90 0 90 0`	
`-1`	

The 23rd Annual

ACM International Collegiate Programming Contest WORLD FINALS

Problem G

The Letter Carrier's Rounds

Input: smtp.in

For an electronic mail application you are to describe the SMTP-based communication that takes place between pairs of MTAs. The sender's User Agent gives a formatted message to the sending Message Transfer Agent (MTA). The sending MTA communicates with the receiving MTA using the Simple Mail Transfer Protocol (SMTP). The receiving MTA delivers mail to the receiver's User Agent. After a communication link is initialized, the sending MTA transmits command lines, one at a time, to the receiving MTA, which returns a three-digit coded response after each command is processed. The sender commands are shown below in the order sent for each message. There is more than one RCPT TO line when the same message is sent to several users at the same MTA. A message to users at different MTAs requires separate SMTP sessions.

`HELO` *myname*	Identifies the sender to the receiver (yes, there is only one L).
`MAIL FROM:`<*sender*>	Identifies the message sender
`RCPT TO:`<*user*>	Identifies one recipient of the message
`DATA`	Followed by an arbitrary number of lines of text comprising the message body, ending with a line containing a period in column one.
`QUIT`	Terminates the communication.

The following response codes are sent by the receiving MTA:

221 Closing connection (after QUIT)
250 Action was okay (after MAIL FROM and RCPT TO specifying an acceptable user, or completion of a message)
354 Start sending mail (after DATA)
550 Action not taken; no such user here (after RCPT TO with unknown user)

Input

The input contains descriptions of MTAs followed by an arbitrary number of messages. Each MTA description begins with the MTA designation and its name (1 to 15 alphanumeric characters). Following the MTA name is the number of users that receive mail at that MTA and a list of the users (1 to 15 alphanumeric characters each). The MTA description is terminated by an asterisk in column 1. Each message begins with the sending user's name and is followed by a list of recipient identifiers. Each identifier has the form *user@mtaname*. The message (each line containing no more than 72 characters) begins and terminates with an asterisk in column 1. A line with an asterisk in column 1 instead of a sender and recipient list indicates the end of the entire input.

Output

For each message, show the communication between the sending and receiving MTAs. Every MTA mentioned in a message is a valid MTA; however, message recipients may not exist at the destination MTA. The receiving MTA rejects mail for those users by responding to the RCPT TO command with the 550 code. A rejection will not affect delivery to authorized users at the same MTA. If there is not at least one authorized recipient at a particular MTA, the DATA is not sent. Only one SMTP session is used to send a message to users at a particular MTA. For example, a message to 5 users at the same MTA will have only one SMTP session. Also a message is addressed to the same user only once. The order in which receiving MTAs are contacted by the sender is unspecified. As shown in the sample output , prefix the communication with the communicating MTA names, and indent each communication line.

Sample Input

```
MTA London 4 Fiona Paul Heather Nevil
MTA SanFrancisco 3 Mario Luigi Shariff
MTA Paris 3 Jacque Suzanne Maurice
MTA HongKong 3 Chen Jeng Hee
```

```
MTA MexicoCity 4 Conrado Estella Eva Raul
MTA Cairo 3 Hamdy Tarik Misa
*
Hamdy@Cairo Conrado@MexicoCity Shariff@SanFrancisco Lisa@MexicoCity
*
Congratulations on your efforts !!
--Hamdy
*
Fiona@London Chen@HongKong Natasha@Paris
*
The reports look great !! --Fiona
*
*
```

Output for the Sample Input

```
Connection between Cairo and MexicoCity
     HELO Cairo
     250
     MAIL FROM:<Hamdy@Cairo>
     250
     RCPT TO:<Conrado@MexicoCity>
     250
     RCPT TO:<Lisa@MexicoCity>
     550
     DATA
     354
     Congratulations on your efforts !!
     --Hamdy
     .
     250
     QUIT
     221
Connection between Cairo and SanFrancisco
     HELO Cairo
     250
     MAIL FROM:<Hamdy@Cairo>
     250
     RCPT TO:<Shariff@SanFrancisco>
     250
     DATA
     354
     Congratulations on your efforts !!
     --Hamdy
     .
     250
     QUIT
     221
Connection between London and HongKong
     HELO London
     250
     MAIL FROM:<Fiona@London>
     250
     RCPT TO:<Chen@HongKong>
     250
     DATA
     354
     Thanks for the report!  --Fiona
     .
     250
     QUIT
     221
Connection between London and Paris
     HELO London
     250
     MAIL FROM:<Fiona@London>
     250
     RCPT TO:<Natasha@Paris>
     550
     QUIT
     221
```

The 23rd Annual

ACM International Collegiate Programming Contest WORLD FINALS

Problem H

Flooded!

Input: water.in

To enable homebuyers to estimate the cost of flood insurance, a real-estate firm provides clients with the elevation of each 10-meter by 10-meter square of land in regions where homes may be purchased. Water from rain, melting snow, and burst water mains will collect first in those squares with the lowest elevations, since water from squares of higher elevation will run downhill. For simplicity, we also assume that storm sewers enable water from high-elevation squares in valleys (completely enclosed by still higher elevation squares) to drain to lower elevation squares, and that water will not be absorbed by the land.

From weather data archives, we know the typical volume of water that collects in a region. As prospective homebuyers, we wish to know the elevation of the water after it has collected in low-lying squares, and also the percentage of the region's area that is completely submerged (that is, the percentage of 10-meter squares whose elevation is strictly less than the water level). You are to write the program that provides these results.

Input

The input consists of a sequence of region descriptions. Each begins with a pair of integers, *m* and *n*, each less than 30, giving the dimensions of the rectangular region in 10-meter units. Immediately following are *m* lines of *n* integers giving the elevations of the squares in row-major order. Elevations are given in meters, with positive and negative numbers representing elevations above and below sea level, respectively. The final value in each region description is an integer that indicates the number of cubic meters of water that will collect in the region. A pair of zeroes follows the description of the last region.

Output

For each region, display the region number (1, 2, …), the water level (in meters above or below sea level) and the percentage of the region's area under water, each on a separate line. The water level and percentage of the region's area under water are to be displayed accurate to two fractional digits. Follow the output for each region with a blank line.

Sample Input	Output for the Sample Input
3 3 25 37 45 51 12 34 94 83 27 10000 0 0	Region 1 Water level is 46.67 meters. 66.67 percent of the region is under water.