Problem A: The Worm Turns

Winston the Worm just woke up in a fresh rectangular patch of earth. The rectangular patch is divided into cells, and each cell contains either food or a rock. Winston wanders aimlessly for a while until he gets hungry; then he immediately eats the food in his cell, chooses one of the four directions (north, south, east, or west) and crawls in a straight line for as long as he can see food in the cell in front of him. If he sees a rock directly ahead of him, or sees a cell where he has already eaten the food, or sees an edge of the rectangular patch, he turns left or right and once again travels as far as he can in a straight line, eating food. He never revisits a cell. After some time he reaches a point where he can go no further so Winston stops, burps and takes a nap.

For instance, suppose Winston wakes up in the following patch of earth (X's represent stones, all other cells contain food):

	0	1	2	3	4
0					X
1					
2					
3		X	X		
4					

If Winston starts eating in row 0, column 3, he might pursue the following path (numbers represent order of visitation):

	0	1	2	3	4
0	4	3	2	1	X
1	5	18	17	16	15
2	6	19	20	21	14
3	7	X	X	22	13
4	8	9	10	11	12

In this case, he chose his path very wisely: every piece of food got eaten. Your task is to help Winston determine where he should begin eating so that his path will visit as many food cells as possible.

Input

Input will consist of multiple test cases. Each test case begins with two positive integers, m and n, defining the number of rows and columns of the patch of earth. Rows and columns are numbered starting at 0, as in the figures above. Following these is a non-negative integer r indicating the number of rocks, followed by a list of $2r$ integers denoting the row and column number of each rock. The last test case is followed by a pair of zeros. This should not be processed. The value $m \times n$ will not exceed 625.

Output

For each test case, print the test case number (beginning with 1), followed by four values:

```
amount row column direction
```

where `amount` is the maximum number of pieces of food that Winston is able to eat, `(row, column)` is the starting location of a path that enables Winston to consume this much food, and `direction` is one of `E, N, S, W`, indicating the initial direction in which Winston starts to move along this path. If there is more than one starting location, choose the one that is lexicographically least in terms of row and column numbers. If there are optimal paths with the same starting location and different starting directions, choose the first valid one in the list `E, N, S, W`. Assume there is always at least one piece of food adjacent to Winston's initial position.

Sample Input

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

Sample Output

```
Case 1: 22 0 3 W
```

Problem B: Jack of All Trades

Jack Barter is a wheeler-dealer of the highest sort. He'll trade anything for anything, as long as he gets a good deal. Recently, he wanted to trade some red agate marbles for some goldfish. Jack's friend Amanda was willing to trade him 1 goldfish for 2 red agate marbles. But Jack did some more digging and found another friend Chuck who was willing to trade him 5 plastic shovels for 3 marbles while Amanda was willing to trade 1 goldfish for 3 plastic shovels. Jack realized that he could get a better deal going through Chuck (1.8 marbles per goldfish) than by trading his marbles directly to Amanda (2 marbles per goldfish).

Jack revels in transactions like these, but he limits the number of other people involved in a chain of transactions to 9 (otherwise things can get a bit out of hand). Normally Jack would use a little program he wrote to do all the necessary calculations to find the optimal deal, but he recently traded away his computer for a fine set of ivory-handled toothpicks. So Jack needs your help.

Input

Input will consist of multiple test cases. The first line of the file will contain an integer n indicating the number of test cases in the file. Each test case will start with a line containing two strings and a positive integer $m \leq 50$. The first string denotes the items that Jack wants, and the second string identifies the items Jack is willing to trade. After this will be m lines of the form

$$a_1 \ \ name_1 \ \ a_2 \ \ name_2$$

indicating that some friend of Jack's is willing to trade an amount a_1 of item $name_1$ for an amount a_2 of item $name_2$. (Note this does not imply the friend is also willing to trade a_2 of item $name_2$ for a_1 of item $name_1$.) The values of a_1 and a_2 will be positive and ≤ 20. No person will ever need more than $2^{31} - 1$ items to complete a successful trade.

Output

For each test case, output the phrase `Case i:` (where `i` is the case number starting at 1) followed by the best possible ratio that Jack can obtain. Output the ratio using 5 significant digits, rounded. Follow this by a single space and then the number of ways that Jack could obtain this ratio.

Sample Input

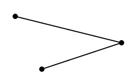
```
2
goldfish marbles 3
1 goldfish 2 marbles
5 shovels 3 marbles
1 goldfish 3 shovels
this that 4
7 this 2 that
14 this 4 that
7 this 2 theother
1 theother 1 that
```

Sample Output

```
Case 1: 1.8000 1
Case 2: 0.28571 3
```

Problem C: Have You Driven a Fjord Lately?

As most of us know, the western Scandinavian coastline contains many small inlets from the sea known as *fjords*. Fjords have very steep sides, and make travel along the coast somewhat tedious (though breathtaking) as the roads must curve back and forth around them. The Fjord Accelerated Scandinavian Traffic Commission (FAST) has decided to solve this problem by putting in a series of bridges across the fjords to cut down on the distances which must be traveled. To save costs, FAST is using pre-constructed bridge units of length 1 meter each, but due to funding restrictions, the total length of bridge that they can build is limited. Therefore, they would like to determine the optimal locations to install bridges that would save the greatest length of road. Fjor example, if a bridge of length 10 meters is built that cuts off 30 meters of old road, a savings of 20 meters is realized. To simplify the determination of where to locate the bridges, FAST has decided to model each fjord as two line segments connecting three points as shown in the figure below.



All the angles making fjords are less than 180°, of course. Furthermore, for safety reasons each bridge can span at most one fjord.

Input

Input for each test case will consist of two lines. The first line contains two positive integers n and m indicating the number of fjords and the maximum length (in meters) of bridge that can be built. The next line will contain $2n + 1$ pairs of integer coordinates for the fjords, where the last coordinate for fjord i serves as the first coordinate for fjord $i + 1$. All coordinates are given in units of meters and will be between -300000 and 300000. The maximum values for n and m are 50 and 3000, respectively. Input will end with the line 0 0.

Output

For each test case output a single line containing the case number followed by the length of the bridge used and the total savings for the optimal placement of bridges, using the format shown below. All values should be in meters and round the latter number to the nearest hundredth.

Sample Input

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

Sample Output

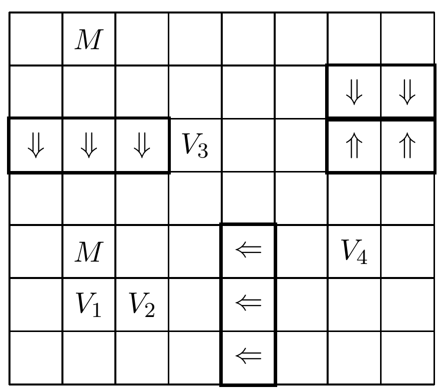
```
Case 1: 6 meters used saving 5.77 meters
Case 2: 6 meters used saving 14.96 meters
Case 3: 8 meters used saving 17.44 meters
```

Problem D: Vampires!

The big media gimmick these days seems to be vampires. Vampire movies, vampire television shows, vampire books, vampire dolls, vampire cereal, vampire lipstick, vampire bunnies – kids and teenagers and even some adults spend lots of time and money on vampire-related stuff. Surprisingly, nowadays vampires are often the good guys. Obviously, the ACM Programming Contest had better have a vampire problem in order to be considered culturally relevant.

As eveyone knows, vampires are allergic to garlic, sunlight, crosses, wooden stakes, and the Internal Revenue Service. But curiously they spend a good part of their time smashing mirrors. Why? Well, mirrors can't hurt vampires physically, but it's embarrassing to be unable to cast a reflection. Mirrors hurt vampire's feelings. This problem is about trying to help them avoid mirrors.

In a room full of vampires and ordinary mortals there are a number of mirrors. Each mirror has one of four orientations – north, south, east, or west (the orientation indicates which side of the mirror reflects). A vampire is in danger of embarrassment if he or she is in a direct horizontal or vertical line with the reflecting side of a mirror, unless there are intervening objects (mortals or other mirrors). For example, in the following room layout



vampire V_2 is exposed to a south-facing mirror and both vampires V_1 and V_2 are exposed to a west-facing mirror (note that a vampire can't protect another vampire from embarrassment since neither one casts a reflection.) Your job is to notify each vampire of the directions in which there is danger of experiencing ENR (embarrassing non-reflectivity).

Input

Each test case begins with three integers v o m, indicating the number of vampires, ordinary mortals, and mirrors in the room. Each of the following v lines contains a pair of integers x, y, $0 \le x, y \le 100$, giving the grid square of a vampire ($x = 0$ corresponds to the westernmost side of the grid and $y = 0$ corresponds to the southernmost side). Each of the following o lines contains the grid square of an ordinary mortal in the same format. Each of the following m lines contains a letter (either N, S, E, or W) indicating the orientation of a mirror, followed by four integers x_1 y_1 x_2 y_2 indicating the start and end squares of the mirror (same bounds as above). Mirrors can be of any positive length, have a thickness of 1 grid square, and will always be aligned along either the east-west or north-south axis. Each grid square contains no more than 1 vampire, mortal, or mirror section. The last test case is followed by a line containing 0 0 0.

Output

For each test case print the case number followed by one line for each vampire that is in danger of embarrassment. On each of these lines, print the vampire's number (vampires are numbered from 1 to v in order of appearance in the input) followed by a list of directions to avoid. For instance, if a vampire is exposed to a south-facing mirror and an east-facing mirror, he/she is exposed in the directions north and west. Directions should be printed in alphabetical order (e.g., `north west`, not `west north`). If no vampires are suffering from embarrassment, output the word `none`. Imitate the sample output.

Sample Input

```
4 2 4
1 1
2 1
3 4
6 2
1 2
1 6
S 0 4 2 4
W 4 2 4 0
N 6 4 7 4
S 6 5 7 5
1 0 2
20 20
W 30 10 30 30
N 25 20 27 20
0 0 0
```

Sample Output

```
Case 1:
vampire 1 east
vampire 2 east north
Case 2:
none
```

Problem E: The Flood

Global warming has us all thinking of rising oceans — well, maybe only those of us who live near the ocean. The small island nation of Gonnasinka has employed you to answer some questions for them. In particular they want to know how high the water has to get before their island becomes two islands (or more).

Given a grid of integers giving the altitudes of the island, how high must the ocean rise before the land splits into pieces?

Input

Each test case begins with a line containing two positive integers n, m giving the dimensions of the igrid, then n lines each containing m positive integers. The integers indicate the original altitude of the grid elements. Grid elements are considered to be adjacent only if they share a horizontal or vertical edge. Values of zero (0) along the perimeter, and all zero cells connected to these, are ocean at its initial level. Cells of 0 not connected to the perimeter (that is, surrounded by higher land) are simply sea level elevations. Furthermore, assume the ocean initially surrounds the given grid. The island is initially connected. Neither n nor m will exceed 100 and heights will never exceed 1000. A line with `0 0` follows the last test case.

Output

For each test case output one of the two following lines.

`Case` n`: Island splits when ocean rises` f `feet.`

or

`Case` n`: Island never splits.`

Our convention here is if your answer is, say, 5 feet, you more accurately mean "5 feet plus a little more." That is, at least a little water will be flowing over the originally 5 foot high portion of land.

Sample Input

```
5 5
3 4 3 0 0
3 5 5 4 3
2 5 4 4 3
1 3 0 0 0
1 2 1 0 0
5 5
5 5 5 5 7
4 1 1 1 4
4 1 2 1 3
7 1 0 0 4
7 3 4 4 4
0 0
```

Sample Output

```
Case 1: Island never splits.
Case 2: Island splits when ocean rises 3 feet.
```