Problem Set for JAG Practice Contest 2010

Japanese Alumni Group

Contest Held on 28 Nov 2010

Status of Problems

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

This problem set was typeset by $\LaTeX\,2_\varepsilon$ with a style file made by a member of JAG from scratch, so that the statement looks like those used in ACM-ICPC Regional Contest held in Japan.

Terms of Use

Problem A
Era Name
Input: A.txt

As many of you know, we have two major calendar systems used in Japan today. One of them is Gregorian calendar which is widely used across the world. It is also known as “Western calendar” in Japan.

The other calendar system is era-based calendar, or so-called “Japanese calendar.” This system comes from ancient Chinese systems. Recently in Japan it has been a common way to associate dates with the Emperors. In the era-based system, we represent a year with an era name given at the time a new Emperor assumes the throne. If the era name is “A”, the first regnal year will be “A 1”, the second year will be “A 2”, and so forth.

Since we have two different calendar systems, it is often needed to convert the date in one calendar system to the other. In this problem, you are asked to write a program that converts western year to era-based year, given a database that contains association between several western years and era-based years.

For the simplicity, you can assume the following:

1. A new era always begins on January 1st of the corresponding Gregorian year.
2. The first year of an era is described as 1.
3. There is no year in which more than one era switch takes place.

Please note that, however, the database you will see may be incomplete. In other words, some era that existed in the history may be missing from your data. So you will also have to detect the cases where you cannot determine exactly which era the given year belongs to.

Input

The input contains multiple test cases. Each test case has the following format:

$$N\ Q$$
$$EraName_1\ EraBasedYear_1\ WesternYear_1$$
$$\vdots$$
$$EraName_N\ EraBasedYear_N\ WesternYear_N$$

$$
\begin{array}{l}
Query_1 \\
\vdots \\
Query_Q
\end{array}
$$

The first line of the input contains two positive integers N and Q ($1 \leq N \leq 1000$, $1 \leq Q \leq 1000$). N is the number of database entries, and Q is the number of queries.

Each of the following N lines has three components: era name, era-based year number and the corresponding western year ($1 \leq EraBasedYear_i \leq WesternYear_i \leq 10^9$). Each of era names consist of at most 16 Roman alphabet characters. Then the last Q lines of the input specifies queries ($1 \leq Query_i \leq 10^9$), each of which is a western year to compute era-based representation.

The end of input is indicated by a line containing two zeros. This line is not part of any dataset and hence should not be processed.

You can assume that all the western year in the input is positive integers, and that there is no two entries that share the same era name.

Output

For each query, output in a line the era name and the era-based year number corresponding to the western year given, separated with a single whitespace. In case you cannot determine the era, output "`Unknown`" without quotes.

Sample Input

```
4 3
meiji 10 1877
taisho 6 1917
showa 62 1987
heisei 22 2010
1868
1917
1988
1 1
universalcentury 123 2168
2010
0 0
```

Output for the Sample Input

```
meiji 1
taisho 6
Unknown
Unknown
```

Problem B
Step Step Evolution
Input: B.txt

Japanese video game company has developed the music video game called Step Step Evolution. The gameplay of Step Step Evolution is very simple. Players stand on the dance platform, and step on panels on it according to a sequence of arrows shown in the front screen.

There are eight types of direction arrows in the Step Step Evolution: UP, UPPER RIGHT, RIGHT, LOWER RIGHT, DOWN, LOWER LEFT, LEFT, and UPPER LEFT. These direction arrows will scroll upward from the bottom of the screen. When the direction arrow overlaps the stationary arrow nearby the top, then the player must step on the corresponding arrow panel on the dance platform. The figure below shows how the dance platform looks like.

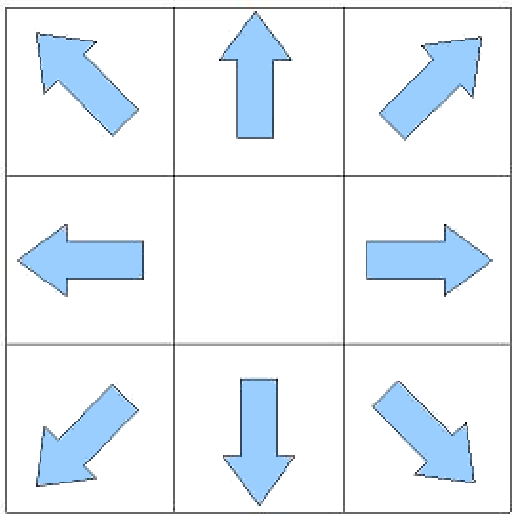


Figure 1: the dance platform for Step Step Evolution

In the game, you have to obey the following rule:

- Except for the beginning of the play, you must not press the arrow panel which is not correspond to the edit data.
- The foot must stay upon the panel where it presses last time, and will not be moved until it's going to press the next arrow panel.

- The left foot must not step on the panel which locates at right to the panel on which the right foot rests. Conversely, the right foot must not step on the panel which locates at left to the panel on which the left foot rests.

As for the third condition, the following figure shows the examples of the valid and invalid footsteps.

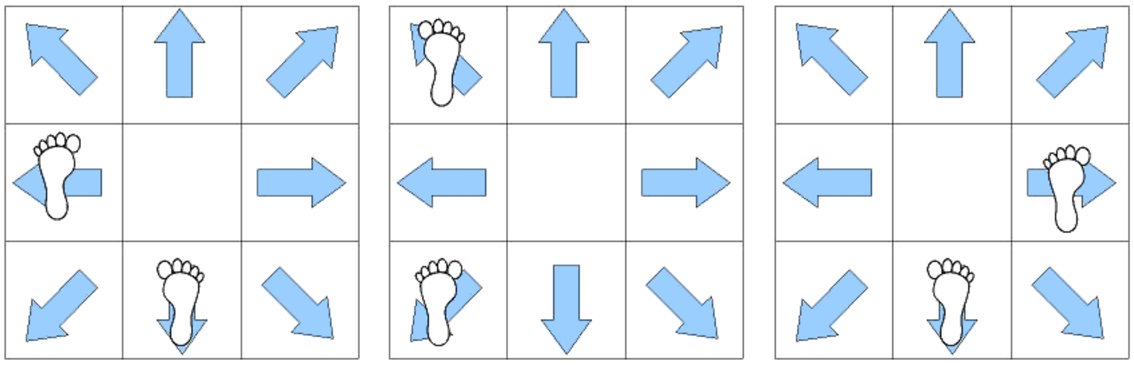


Figure 2: the examples of valid and invalid footsteps

The first one (left foot for LEFT, right foot for DOWN) and the second one (left foot for LOWER LEFT, right foot for UPPER LEFT) are valid footstep. The last one (left foot for RIGHT, right foot for DOWN) is invalid footstep.

Note that, at the beginning of the play, the left foot and right foot can be placed anywhere at the arrow panel. Also, you can start first step with left foot or right foot, whichever you want.

To play this game in a beautiful way, the play style called " Natural footstep style" is commonly known among talented players. "Natural footstep style" is the style in which players make steps by the left foot and the right foot in turn. However, when a sequence of arrows is difficult, players are sometimes forced to violate this style.

Now, your friend has created an edit data (the original sequence of direction arrows to be pushed) for you. You are interested in how many times you have to violate "Natural footstep style" when you optimally played with this edit data. In other words, what is the minimum number of times you have to step on two consecutive arrows using the same foot?

Input

The input consists of several detasets. Each dataset is specified by one line containing a sequence of direction arrows. Directions are described by numbers from 1 to 9, except for 5. The figure below shows the correspondence between numbers and directions.

You may assume that the length of the sequence is between 1 and 100000, inclusive. Also, arrows of the same direction won't appear consecutively in the line. The end of the input is indicated by a single "#".

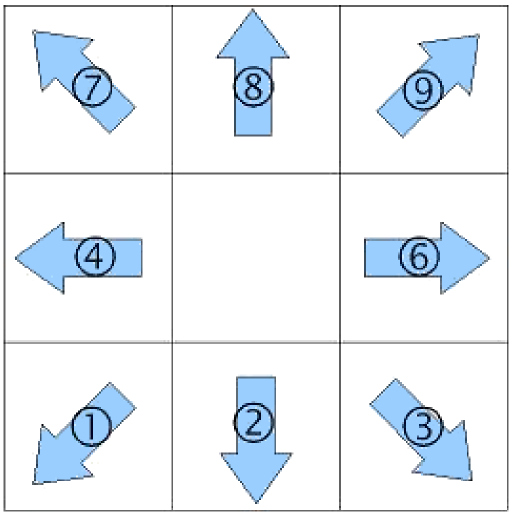


Figure 3: the correspondence of the numbers to the direction arrows

Output

For each dataset, output how many times you have to violate “Natural footstep style” when you play optimally in one line.

Sample Input

```
1
12
123
369
6748
4247219123
1232123212321232
#
```

Output for the Sample Input

```
0
0
1
0
1
2
7
```

Problem C
Dungeon Quest II
Input: C.txt

The cave, called "Mass of Darkness", had been a agitating point of the evil, but the devil king and all of his soldiers were destroyed by the hero and the peace is there now.

One day, however, the hero was worrying about the rebirth of the devil king, so he decided to ask security agency to patrol inside the cave.

The information of the cave is as follows:

- The cave is represented as a two-dimensional field which consists of rectangular grid of cells.
- The cave has $R \times C$ cells where R is the number of rows and C is the number of columns.
- Some of the cells in the cave contains a trap, and those who enter the trapping cell will lose his hit points.
- The type of traps varies widely: some of them reduce hit points seriously, and others give less damage.

The following is how the security agent patrols:

- The agent will start his patrol from upper left corner of the cave.
  - There are no traps at the upper left corner of the cave.
- The agent will patrol by tracing the steps which are specified by the hero.
  - The steps will be provided such that the agent never go outside of the cave during his patrol.
- The agent will bring potions to regain his hit point during his patrol.
- The agent can use potions just before entering the cell where he is going to step in.
- The type of potions also varies widely: some of them recover hit points so much, and others are less effective.
  - Note that agent's hit point can be recovered up to HP_{max} which means his maximum hit point and is specified by the input data.

- The agent can use more than one type of potion at once.
- If the agent's hit point becomes less than or equal to 0, he will die.

Your task is to write a program to check whether the agent can finish his patrol without dying.

Input

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

> HP_{init} HP_{max}
> R C
> $a_{1,1}$ $a_{1,2}$ $\cdots$ $a_{1,C}$
> $a_{2,1}$ $a_{2,2}$ $\cdots$ $a_{2,C}$
> $\vdots$
> $a_{R,1}$ $a_{R,2}$ $\cdots$ $a_{R,C}$
> T
> `[A-Z]` d_1
> `[A-Z]` d_2
> $\vdots$
> `[A-Z]` d_T
> S
> `[UDLR]` n_1
> `[UDLR]` n_2
> $\vdots$
> `[UDLR]` n_S
> P
> p_1
> p_2
> $\vdots$
> p_P

The first line of a dataset contains two integers HP_{init} and HP_{max} ($0 < HP_{init} \leq HP_{max} \leq 1000$), meaning the agent's initial hit point and the agent's maximum hit point respectively.

The next line consists of R and C ($1 \leq R, C \leq 100$). Then, R lines which made of C characters representing the information of the cave follow. The character $a_{i,j}$ means there are the trap of type $a_{i,j}$ in i-th row and j-th column, and the type of trap is denoted as an uppercase alphabetic character `[A-Z]`.

The next line contains an integer T, which means how many type of traps to be described. The following T lines contains a uppercase character `[A-Z]` and an integer d_i ($0 \leq d_i \leq 1000$), representing the type of trap and the amount of damage it gives.

The next line contains an integer S ($0 \le S \le 1000$) representing the number of sequences which the hero specified as the agent's patrol route. Then, S lines follows containing a character and an integer n_i ($\sum_{i=1}^{S} n_i \le 1000$), meaning the direction where the agent advances and the number of step he takes toward that direction. The direction character will be one of '`U`', '`D`', '`L`', '`R`' for Up, Down, Left, Right respectively and indicates the direction of the step.

Finally, the line which contains an integer P ($0 \le P \le 12$) meaning how many type of potions the agent has follows. The following P lines consists of an integer p_i ($0 < p_i \le 1000$) which indicated the amount of hit point it recovers.

The input is terminated by a line with two zeros. This line should not be processed.

Output

For each dataset, print in a line "`YES`" if the agent finish his patrol successfully, or "`NO`" otherwise.

If the agent's hit point becomes less than or equal to 0 at the end of his patrol, the output should be "`NO`".

Sample Input

```
1 10
3 3
AAA
ABA
CCC
3
A 0
B 5
C 9
3
D 2
R 1
U 2
5
10
10
10
10
10
100 100
10 10
THISISAPEN
THISISAPEN
THISISAPEN
THISISAPEN
```

```
THISISAPEN
THISISAPEN
THISISAPEN
THISISAPEN
THISISAPEN
THISISAPEN
8
T 0
H 1
I 2
S 3
A 4
P 5
E 6
N 7
9
R 1
D 3
R 8
D 2
L 9
D 2
R 9
D 2
L 9
2
20
10
0 0
```

Output for the Sample Input

```
YES
NO
```

Problem D
Dice Room
Input: D.txt

You are playing a popular video game which is famous for its depthful story and interesting puzzles. In the game you were locked in a mysterious house alone and there is no way to call for help, so you have to escape on yours own. However, almost every room in the house has some kind of puzzles and you cannot move to neighboring room without solving them.

One of the puzzles you encountered in the house is following. In a room, there was a device which looked just like a dice and laid on a table in the center of the room. Direction was written on the wall. It read:

> "This cube is a remote controller and you can manipulate a remote room, Dice Room, by it. The room has also a cubic shape whose surfaces are made up of 3x3 unit squares and some squares have a hole on them large enough for you to go though it. You can rotate this cube so that in the middle of rotation at least one edge always touch the table, that is, to 4 directions. Rotating this cube affects the remote room in the same way and positions of holes on the room change. To get through the room, you should have holes on at least one of lower three squares on the front and back side of the room."

You can see current positions of holes by a monitor. Before going to Dice Room, you should rotate the cube so that you can go though the room. But you know rotating a room takes some time and you don't have much time, so you should minimize the number of rotation. How many rotations do you need to make it possible to get though Dice Room?

Input

The input consists of several datasets. Each dataset contains 6 tables of 3x3 characters which describe the initial position of holes on each side. Each character is either '`*`' or '`.`'. A hole is indicated by '`*`'. The order which six sides appears in is: front, right, back, left, top, bottom. The order and orientation of them are described by this development view:

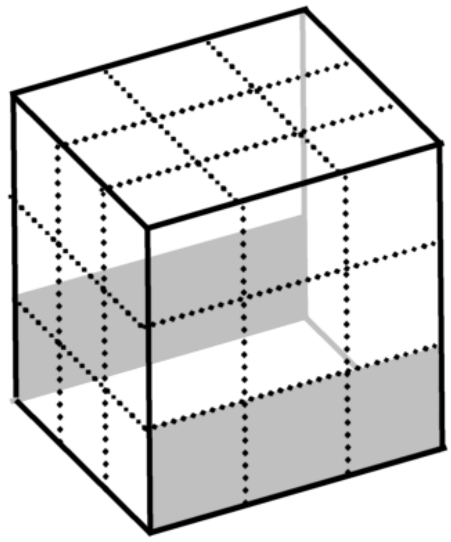


Figure 4: Dice Room

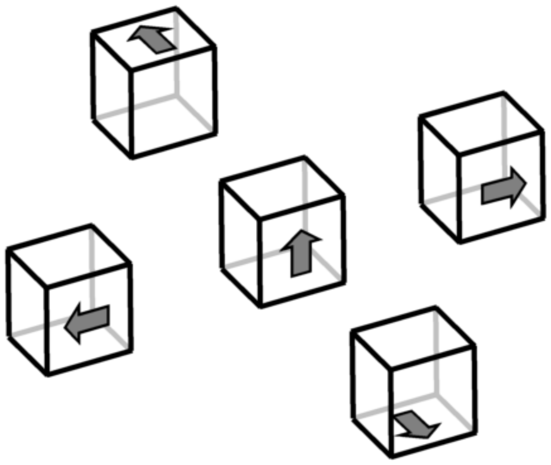
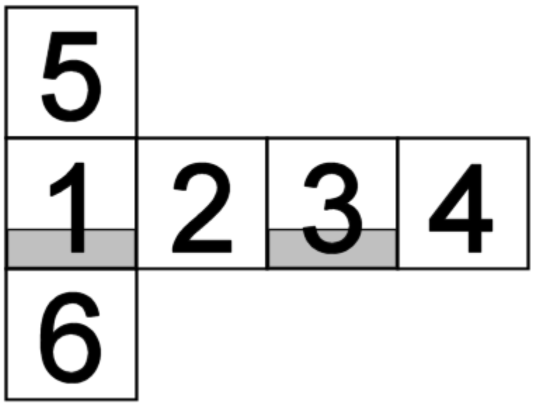


Figure 5: available dice rotations



There is a blank line after each dataset. The end of the input is indicated by a single '#'.

Output

Print the minimum number of rotations needed in a line for each dataset. You may assume all datasets have a solution.

Sample Input

```
...
...
.*.
...
...
.*.
...
...
...
...
...
.*.
...
...
.*.
...
...
...

...
.*.
```

```
...
*..
...
..*
*.*
*.*
*.*
*.*
.*.
*.*
*..
.*.
..*
*.*
...
*.*

...
.*.
.*.
...
.**
*..
...
...
.*.
.*.
...
*..
..*
...
.**
...
*..
...

#
```

Output for the Sample Input

```
3
1
0
```

Problem E
Alice and Bomb
Input: E.txt

Alice and Bob were in love with each other, but they hate each other now.

One day, Alice found a bag. It looks like a bag that Bob had used when they go on a date. Suddenly Alice heard tick-tack sound. Alice intuitively thought that Bob is to kill her with a bomb. Fortunately, the bomb has not exploded yet, and there may be a little time remained to hide behind the buildings.

The appearance of the ACM city can be viewed as an infinite plane and each building forms a polygon. Alice is considered as a point and the bomb blast may reach Alice if the line segment which connects Alice and the bomb does not intersect an interior position of any building. Assume that the speed of bomb blast is infinite; when the bomb explodes, its blast wave will reach anywhere immediately, unless the bomb blast is interrupted by some buildings.

The figure below shows the example of the bomb explosion. Left figure shows the bomb and the buildings(the polygons filled with black). Right figure shows the area(colored with gray) which is under the effect of blast wave after the bomb explosion.

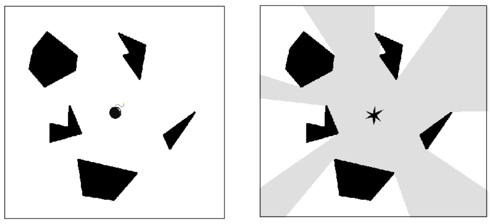


Figure 6: The example of the bomb explosion

Note that even if the line segment connecting Alice and the bomb touches a border of a building, bomb blast still can blow off Alice. Since Alice wants to escape early, she wants to minimize the length she runs in order to make herself hidden by the building.

Your task is to write a program which reads the positions of Alice, the bomb and the buildings and calculate the minimum distance required for Alice to run and hide behind the building.

Input

The input contains multiple test cases. Each test case has the following format:

N
$bx\ by$
$m_1\ x_{1,1}\ y_{1,1}\ \dots\ x_{1,m_1}\ y_{1,m_1}$
$\vdots$
$m_N\ x_{N,1}\ y_{N,1}\ \dots\ x_{N,m_N}\ y_{N,m_N}$

The first line of each test case contains an integer N ($1 \leq N \leq 100$), which denotes the number of buildings. In the second line, there are two integers bx and by ($-10000 \leq bx, by \leq 10000$), which means the location of the bomb. Then, N lines follows, indicating the information of the buildings.

The information of building is given as a polygon which consists of points. For each line, it has an integer m_i ($3 \leq m_i \leq 100$, $\sum_{i=1}^{N} m_i \leq 500$) meaning the number of points the polygon has, and then m_i pairs of integers $x_{i,j}$, $y_{i,j}$ follow providing the x and y coordinate of the point.

You can assume that

- the polygon does not have self-intersections.
- any two polygons do not have a common point.
- the set of points is given in the counter-clockwise order.
- the initial positions of Alice and bomb will not by located inside the polygon.
- there are no bomb on the any extended lines of the given polygon's segment.

Alice is initially located at (0, 0). It is possible that Alice is located at the boundary of a polygon.

$N = 0$ denotes the end of the input. You may not process this as a test case.

Output

For each test case, output one line which consists of the minimum distance required for Alice to run and hide behind the building. An absolute error or relative error in your answer must be less than 10^{-6}.

Sample Input

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

```
4 1 1 1 2 -1 2 -1 1
1
-6 -6
6 1 -2 2 -2 2 3 -2 3 -2 1 1 1
1
-10 0
4 0 -5 1 -5 1 5 0 5
1
10 1
4 5 1 6 2 5 3 4 2
2
-47 -37
4 14 3 20 13 9 12 15 9
4 -38 -3 -34 -19 -34 -14 -24 -10
0
```

Output for the Sample Input

```
1.00000000
0.00000000
3.23606798
5.00000000
1.00000000
11.78517297
```

Problem F
Two-Wheel Buggy
Input: F.txt

International Car Production Company (ICPC), one of the largest automobile manufacturers in the world, is now developing a new vehicle called "Two-Wheel Buggy". As its name suggests, the vehicle has only two wheels. Quite simply, "Two-Wheel Buggy" is made up of two wheels (the left wheel and the right wheel) and a axle (a bar connecting two wheels). The figure below shows its basic structure.

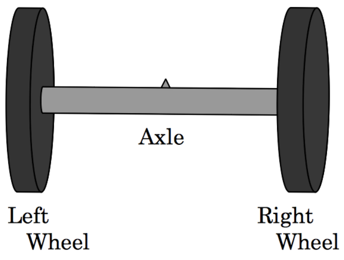


Figure 7: The basic structure of the buggy

Before making a prototype of this new vehicle, the company decided to run a computer simulation. The details of the simulation is as follows.

In the simulation, the buggy will move on the x-y plane. Let D be the distance from the center of the axle to the wheels. At the beginning of the simulation, the center of the axle is at $(0, 0)$, the left wheel is at $(-D, 0)$, and the right wheel is at $(D, 0)$. The radii of two wheels are 1.

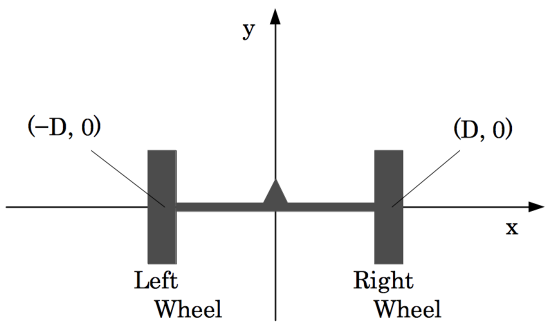


Figure 8: The initial position of the buggy

The movement of the buggy in the simulation is controlled by a sequence of instructions. Each instruction consists of three numbers, *Lspeed*, *Rspeed* and *time*. *Lspeed* and *Rspeed* indicate the rotation speed of the left and right wheels, respectively, expressed in degree par second. *time* indicates how many seconds these two wheels keep their rotation speed. If a speed of a wheel is positive, it will rotate in the direction that causes the buggy to move forward. Conversely, if a speed is negative, it will rotate in the opposite direction. For example, if we set *Lspeed* as -360, the left wheel will rotate 360-degree in one second in the direction that makes the buggy move backward. We can set *Lspeed* and *Rspeed* differently, and this makes the buggy turn left or right. Note that we can also set one of them positive and the other negative (in this case, the buggy will spin around).

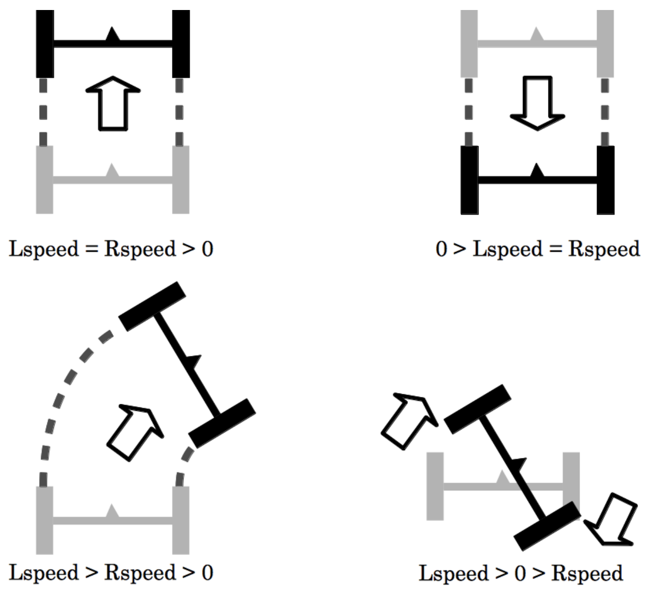


Figure 9: Examples

Your job is to write a program that calculates the final position of the buggy given a instruction sequence. For simplicity, you can can assume that wheels have no width, and that they would never slip.

Input

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

N D
$Lspeed_1$ $Rspeed_1$ $time_1$
$\vdots$
$Lspeed_i$ $Rspeed_i$ $time_i$
$\vdots$
$Lspeed_N$ $Rspeed_N$ $time_N$

The first line of a dataset contains two positive integers, N and D ($1 \leq N \leq 100$, $1 \leq D \leq 10$). N indicates the number of instructions in the dataset, and D indicates the distance between the center of axle and the wheels. The following N lines describe the instruction sequence. The i-th line contains three integers, $Lspeed_i$, $Rspeed_i$, and $time_i$ ($-360 \leq Lspeed_i, Rspeed_i \leq 360$, $1 \leq time_i$), describing the i-th instruction to the buggy. You can assume that the sum of $time_i$ is at most 500.

The end of input is indicated by a line containing two zeros. This line is not part of any dataset and hence should not be processed.

Output

For each dataset, output two lines indicating the final position of the center of the axle. The first line should contain the x-coordinate, and the second line should contain the y-coordinate. The absolute error should be less than or equal to 10^{-3}. No extra character should appear in the output.

Sample Input

```
1 1
180 90 2
1 1
180 180 20
2 10
360 -360 5
-90 360 8
3 2
100 60 9
-72 -72 10
-45 -225 5
0 0
```

Output for the Sample Input

```
3.00000
3.00000
0.00000
62.83185
12.00000
0.00000
-2.44505
13.12132
```

Problem G
Camera Control
Input: G.txt

"ACM48" is one of the most popular dance vocal units in Japan. In this winter, ACM48 is planning a world concert tour. You joined the tour as a camera engineer.

Your role is to develop software which controls the camera on a stage. For simplicity you can regard the stage as 2-dimensional space. You can rotate the camera to an arbitrary direction by the software but cannot change its coordinate.

During a stage performance, each member of ACM48 moves along her *route* and sings the part(s) assigned to her. Here, a route is given as a polygonal line.

You have to keep focusing the camera on a member during a stage performance. You can change the member focused by the camera if and only if the current and next members are in the same direction from the camera.

Your task is to write a program which reads the stage performance plan and calculates the maximum time that you can focus the camera on members that are singing.

You may assume the following are satisfied:

- You can focus the camera on an arbitrary member at the beginning time.
- Each route of the member does not touch the camera.
- Each member stays at the last coordinates after she reaches there.

Input

The input contains multiple test cases. Each test case has the following format:

> N
> c_x c_y
> The information of the 1-st member
> $\vdots$
> The information of the N-th member

N ($1 \leq N \leq 50$) is the number of the members. (c_x, c_y) is the coordinates of the camera. Then the information of the N members follow.

The information of the i-th member has the following format:

$$\begin{array}{l} M_i \\ x_{i,1}\ y_{i,1}\ t_{i,1} \\ \vdots \\ x_{i,M_i}\ y_{i,M_i}\ t_{i,M_i} \\ L_i \\ b_{i,1}\ e_{i,1} \\ \vdots \\ b_{i,L_i}\ e_{i,L_i} \end{array}$$

M_i ($1 \leq M_i \leq 100$) is the number of the points in the route. ($x_{i,j}$, $y_{i,j}$) is the coordinates of the j-th in the route. $t_{i,j}$ ($0 = t_{i,0} < t_{i,j} < t_{i,j+1} \leq 10^3$ for $0 < j$) is the time that the i-th member reaches the j-th coordinates. L_i ($0 \leq L_i \leq 100$) is the number of the vocal part. $b_{i,k}$ and $e_{i,k}$ ($0 \leq b_{i,k} < e_{i,k} < b_{i,k+1} < e_{i,k+1} \leq 10^3$) are the beginning and the ending time of the k-th vocal part, respectively.

All the input values are integers. You may assume that the absolute of all the coordinates are not more than 10^3.

$N = 0$ denotes the end of the input. You may not process this as a test case.

Output

For each dataset, print the maximum time that you can focus the camera on singing members with an absolute error of at most 10^{-6}. You may output any number of digits after the decimal point.

Sample Input

```
2
0 0
2
-5 5 0
5 5 10
1
0 6
2
5 5 0
-5 5 10
1
6 10
1
7 -65
```

```
2
-65 10 0
65 1 3
2
0 1
23 24
2
0 0
2
100 10 0
-10 10 10
5
0 1
2 3
4 5
6 7
8 9
2
10 0 0
0 10 10
5
1 2
3 4
5 6
7 8
9 10
0
```

Output for the Sample Input

```
9.00000000
2.00000000
5.98862017
```

Problem H
Road Construction
Input: H.txt

King Mercer is the king of ACM kingdom. There are one capital and some cities in his kingdom. Amazingly, there are no roads in the kingdom now. Recently, he planned to construct roads between the capital and the cities, but it turned out that the construction cost of his plan is much higher than expected.

In order to reduce the cost, he has decided to create a new construction plan by removing some roads from the original plan. However, he believes that a new plan should satisfy the following conditions:

- For every pair of cities, there is a route (a set of roads) connecting them.
- The minimum distance between the capital and each city does not change from his original plan.

Many plans may meet the conditions above, but King Mercer wants to know the plan with minimum cost. Your task is to write a program which reads his original plan and calculates the cost of a new plan with the minimum cost.

Input

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

N M
u_1 v_1 d_1 c_1
$\vdots$
u_M v_M d_M c_M

The first line of each dataset begins with two integers, N and M ($1 \le N \le 10000$, $0 \le M \le 20000$). N and M indicate the number of cities and the number of roads in the original plan, respectively.

The following M lines describe the road information in the original plan. The i-th line contains four integers, u_i, v_i, d_i and c_i ($1 \le u_i, v_i \le N$, $u_i \ne v_i$, $1 \le d_i \le 1000$, $1 \le c_i \le 1000$). u_i, v_i, d_i and c_i indicate that there is a road which connects u_i-th city and v_i-th city, whose length is d_i and whose cost needed for construction is c_i.

Each road is bidirectional. No two roads connect the same pair of cities. The 1-st city is the capital in the kingdom.

The end of the input is indicated by a line containing two zeros separated by a space. You should not process the line as a dataset.

Output

For each dataset, print the minimum cost of a plan which satisfies the conditions in a line.

Sample Input

```
3 3
1 2 1 2
2 3 2 1
3 1 3 2
5 5
1 2 2 2
2 3 1 1
1 4 1 1
4 5 1 1
5 3 1 1
5 10
1 2 32 10
1 3 43 43
1 4 12 52
1 5 84 23
2 3 58 42
2 4 86 99
2 5 57 83
3 4 11 32
3 5 75 21
4 5 23 43
5 10
1 2 1 53
1 3 1 65
1 4 1 24
1 5 1 76
2 3 1 19
2 4 1 46
2 5 1 25
3 4 1 13
3 5 1 65
4 5 1 34
0 0
```

Output for the Sample Input

```
3
5
137
218
```

Problem I
Operator
Input: I.txt

The customer telephone support center of the computer sales company called JAG is now incredibly confused. There are too many customers who request the support, and they call the support center all the time. So, the company wants to figure out how many operators needed to handle this situation.

For simplicity, let us focus on the following simple simulation.

Let N be a number of customers. The i-th customer has id i, and is described by three numbers, M_i, L_i and K_i. M_i is the time required for phone support, L_i is the maximum stand by time until an operator answers the call, and K_i is the interval time from hanging up to calling back. Let us put these in other words: It takes M_i unit times for an operator to support i-th customer. If the i-th customer is not answered by operators for L_i unit times, he hangs up the call. K_i unit times after hanging up, he calls back.

One operator can support only one customer simultaneously. When an operator finish a call, he can immediately answer another call. If there are more than one customer waiting, an operator will choose the customer with the smallest id.

At the beginning of the simulation, all customers call the support center at the same time. The simulation succeeds if operators can finish answering all customers within T unit times.

Your mission is to calculate the minimum number of operators needed to end this simulation successfully.

Input

The input contains multiple datasets. Each dataset has the following format:

$$\begin{array}{l} N\ T \\ M_1\ L_1\ K_1 \\ \vdots \\ M_N\ L_N\ K_N \end{array}$$

The first line of a dataset contains two positive integers, N and T ($1 \le N \le 1000$, $1 \le T \le 1000$). N indicates the number of customers in the dataset, and T indicates the time limit of the simulation.

The following N lines describe the information of customers. The i-th line contains three integers, M_i, L_i and K_i ($1 \leq M_i \leq T$, $1 \leq L_i \leq 1000$, $1 \leq K_i \leq 1000$), describing i-th customer's information. M_i indicates the time required for phone support, L_i indicates the maximum stand by time until an operator answers the call, and K_i indicates the is the interval time from hanging up to calling back.

The end of input is indicated by a line containing two zeros. This line is not part of any dataset and hence should not be processed.

Output

For each dataset, print the minimum number of operators needed to end the simulation successfully in a line.

Sample Input

```
3 300
100 50 150
100 50 150
100 50 150
3 300
100 50 150
100 50 150
200 50 150
9 18
3 1 1
3 1 1
3 1 1
4 100 1
5 100 1
5 100 1
10 5 3
10 5 3
1 7 1000
10 18
1 2 3
2 3 4
3 4 5
4 5 6
5 6 7
6 7 8
7 8 9
8 9 10
9 10 11
10 11 12
```

```
0 0
```

Output for the Sample Input

```
2
3
3
4
```

Problem J
Merry Christmas
Input: J.txt

International Christmas Present Company (ICPC) is a company to employ Santa and deliver presents on Christmas. Many parents request ICPC to deliver presents to their children at specified time of December 24. Although same Santa can deliver two or more presents, because it takes time to move between houses, two or more Santa might be needed to finish all the requests on time.

Employing Santa needs much money, so the president of ICPC employed you, a great programmer, to optimize delivery schedule. Your task is to write a program to calculate the minimum number of Santa necessary to finish the given requests on time. Because each Santa has been well trained and can conceal himself in the town, you can put the initial position of each Santa anywhere.

Input

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

$$\begin{array}{l} N\ M\ L \\ u_1\ v_1\ d_1 \\ u_2\ v_2\ d_2 \\ \vdots \\ u_M\ v_M\ d_M \\ p_1\ t_1 \\ p_2\ t_2 \\ \vdots \\ p_L\ t_L \end{array}$$

The first line of a dataset contains three integer, N, M and L ($1 \leq N \leq 100$, $0 \leq M \leq 1000$, $1 \leq L \leq 1000$) each indicates the number of houses, roads and requests respectively.

The following M lines describe the road network. The i-th line contains three integers, u_i, v_i, and d_i ($0 \leq u_i < v_i \leq N-1$, $1 \leq d_i \leq 100$) which means that there is a road connecting houses u_i and v_i with d_i length. Each road is bidirectional. There is at most one road between same pair of houses. Whole network might be disconnected.

The next L lines describe the requests. The i-th line contains two integers, p_i and t_i ($0 \leq p_i \leq N-1$, $0 \leq t_i \leq 10^8$) which means that there is a delivery request to house p_i on time t_i. There is at most one request for same place and time. You can assume that time taken other than movement can be neglectable, and every Santa has the same speed, one unit distance per unit time.

The end of the input is indicated by a line containing three zeros separated by a space, and you should not process this as a test case.

Output

Print the minimum number of Santa necessary to finish all the requests on time.

Sample Input

```
3 2 3
0 1 10
1 2 10
0 0
1 10
2 0
3 2 4
0 1 10
1 2 10
0 0
1 10
2 20
0 40
10 10 10
0 1 39
2 3 48
3 5 20
4 8 43
3 9 10
8 9 40
3 4 5
5 7 20
1 7 93
1 3 20
0 0
1 100000000
2 100
3 543
4 500
5 400
```

```
6 300
7 200
8 100
9 100
0 0 0
```

Output for the Sample Input

```
2
1
4
```