Problem A - BEATBIT

Background

A software company is very much concerned that its software engineers do not write equivalent procedures into the new version of its main product BEATBIT. The BEATBIT system is written in the assembly language BITE, recently introduced by MacroSoft, in its .DOT Framework. So far, no one has invented a more powerful language to program complex digital circuits. The BITE assembly language operates on a stack of bits, and is defined by the following four kinds of instructions:

> *label* **BRTRUE** *destlabel*
> Pops a bit from the top of the stack, and tests it. If the value is **1** continues execution at the instruction with label *destlabel*. If the value is **0** continues at the next program instruction.
>
> *label* **JMP** *destlabel*
> Continues execution at the instruction with label *destlabel*.
>
> *label* **RET1**
> Stops execution and returns **1**.
>
> *label* **RET0**
> Stops execution and returns **0**.

Here, *label* and *destlabel* are positive integers. A *n-ary* procedure of BITE is a sequence of instructions that expects **N** bit values on the stack as input, and produces one bit value, as the result of a **RET0** or **RET1** instruction. The instructions in the sequence are always labelled in increasing label sequence, and it is known that, for every possible input, the procedures always terminate. The program starts at the instruction with the lowest label.

Problem

Write a program that checks whether two BITE procedures compute the same boolean function for any sequence of the values, stored in the stack, provided as input.

Input

A positive integer **P** in a single line followed by a sequence of **P** pairs of BITE procedures. Each pair of BITE procedures is preceded by the number of bits expected in the stack (the arity of the procedures), represented by a positive integer, not greater than 128, in a single line. Next, for each BITE procedure there is a sequence of lines, each one containing a BITE instruction, and terminated by a single line containing **`END`**. Programs do not have more than 10,000 lines of code.

Output

A sequence of lines, the *i-th* line containing either **1** or **0** depending on whether the *i−th* pair of BITE procedures in the input compute the same boolean function or not.

Sample Input

```
2
2
10 BRTRUE 30
20 RET0
30 BRTRUE 50
40 RET0
50 RET1
END
20 BRTRUE 50
30 BRTRUE 40
40 RET0
50 BRTRUE 70
60 JMP 40
70 RET1
END
1
10 BRTRUE 30
20 RET0
30 RET1
END
50 RET0
END
```

Sample Output

```
1
0
```

Problem B - Prester John

Background

Where was the Kingdom of Prester John? There are many maps showing the way to it, but unfortunately it seems hard to establish a sensible agreement. India? Ethiopia? Mongolia? Syria? Puzzling… Well, what would one expect from the descendant of the Three Wise Men?
An idea is to follow the directions in two maps at the same time, and if the directions lead to the same place, we can find Prester. Well, "information technology" may help here. Write a program that, given two medieval maps of the world indicating the location of the Mythical Kingdom, finds the length of the common shortest path to the location where the Prester was seen.

Problem

Given two maps, compute the minimum length of a common path to the Prester location.

Input

The input contains the description of a pair of maps, in sequence. Each map is given by a positive integer **L**, not greater than 50,000, in a single line, indicating the number of locations in the map. Next, there is an integer in the range **[0, L-1]** indicating the location, in a single line, where the Prester was seen. Next, there is a positive integer **P**, not greater than 100,000, in a single line, indicating the number of paths in the description of the map. Then, the description of the map follows. Each path in the map is listed in a separate line, and has the form

L1 description L2

where *L1* and *L2* are integers in the range **[0, L-1]** indicating a location, and *description* is a string with no more than 8 characters, indicating the name of a path from location *L1* to location *L2*. It is known that location **0** represents the same place in all maps.

Output

An integer in a single line indicating the length of the shortest sequence of path descriptions that is common to both maps, and that, in both maps, lead to a location of the Prester. If there is no common path leading to the Prester location, your program should write **-1** as result.

Sample Input

```
2
1
2
0 tunnel 1
1 bridge 1
3
2
3
0 tunnel 1
1 bridge 2
2 river 2
```

Sample Output

```
2
```

Problem C - Robotruck

Background

This problem is about a robotic truck that distributes mail packages to several locations in a factory. The robot sits at the end of a conveyer at the mail office and waits for packages to be loaded into its cargo area. The robot has a maximum load capacity, which means that it may have to perform several round trips to complete its task. Provided that the maximum capacity is not exceeded, the robot can stop the conveyer at any time and start a round trip distributing the already collected packages. The packages must be delivered in the incoming order.
The distance of a round trip is computed in a grid by measuring the number of robot moves from the mail office, at location (0,0), to the location of delivery of the first package, the number of moves between package delivery locations, until the last package, and then the number of moves from the last location back to the mail office. The robot moves a cell at a time either horizontally or vertically in the factory plant grid. For example, consider four packages, to be delivered at the locations (1,2), (1,0), (3,1), and (3,1). By dividing these packages into two round trips of two packages each, the number of moves in the first trip is 3+2+1=6, and 4+0+4=8 in the second trip. Notice that the two last packages are delivered at the same location and thus the number of moves between them is 0.

Problem

Given a sequence of packages, compute the minimum distance the robot must travel to deliver all packages.

Input

The input consists of a line containing one positive integer indicating the maximum capacity of the robot, a line containing one positive integer **N**, not greater than 100,000, which is the number of packages to be loaded from the conveyer. Next, there are **N** lines containing, for each package, two non-negative integers to indicate its delivery location in the grid, and a non-negative integer to indicate its weight. The weight of the packages is always smaller than the robot's maximum load capacity. The order of the input is the order of appearance in the conveyer.

Output

One line containing one integer representing the minimum number of moves the robot must travel to deliver all the packages.

Sample Input

```
10
4
1 2 3
1 0 3
3 1 4
3 1 4
```

Sample Output

```
14
```

Problem D - Jumping Hero

Background

A software house has decided to create a computer game, where the hero must find its way from a start position to the end position, through a labyrinth. In the labyrinth, some cells contain magic fountains that can be used to get super-powers an infinite number of times. Whenever the hero enters a cell with a magic fountain, he gets super-powers.

Usually, our hero moves in the labyrinth one cell left/right/up/down at a time (to an empty cell). With super-powers, the hero jumps to an empty cell N positions to the left/right/up/down. The super-power lasts for M jumps, and the hero can change its jumping direction after each jump. A jump is allowed if the end cell of the jump is within the map and it is not a wall â€“ thus, the hero can jump over walls. If the hero jumps to a cell with a new magic fountain, the hero gets the super-powers of the new magic fountain, and the remaining effect of the previous magic fountain is cancelled. If the hero jumps to the cell where he obtained its current super-powers, no effect occurs (i.e., the hero gets no additional super-powers). When the current super-power ends, the hero proceeds its normal one-cell movement. If, after getting super-powers in some fountain, the hero cannot move to any cell, he looses his super-powers and returns to his previous cell. To reach the end position, the hero must move to the end cell or finish one jump in the end cell.

Problem

Given the labyrinth map compute the minimum number of moves/jumps from the start position to the end position.

Input

The first line of the input contains two positive integers, L and C, separated by a empty space, with L the number of lines and C the number of columns in the map. L and C are both lesser than 300. The following L lines of the input contain C integers each that define the cells of the map (separated by a empty space). Each integer, i, must be interpreted as follows: i = 0 represents a wall; i = 1 represents an empty cell (where the hero can move into); i = M*10+N represents an empty cell with a magic fountain that makes the hero jump M times to the cell that is N positions to the left/right/up/down of the current cell. M ranges from 1 to 5 and N ranges from 2 to 6. The maximum number of magic fountains in a map is 5,000. The two last lines of the input define the coordinates of the start position and end position (coordinates consist of two integers, denoting the line and column respectively, starting from 0).

Output

The output consists of one single line that contains an integer with the minimum number of moves/jumps, from the start position to the end position. If it is impossible to reach the end position, the output should be a single line containing `IMPOSSIBLE`.

Sample Input

```
8 8
0 1 1 1 1 1 1 1
0 1 0 0 1 13 1 1
0 1 32 1 1 1 0 0
```

```
0 1 1 0 1 1 1 0
0 1 1 0 0 0 0 0
0 1 1 1 1 1 1 0
0 1 0 0 1 1 1 0
0 1 1 1 1 1 1 0
1 7
5 4
```

Sample Output

```
14
```

Problem E - Board Games

Background

You have been hired by the quality control division of a world famous board game company. Their creative and design division comes up, on a daily basis, with great ideas for board games, but sometimes the scoring of the proposed games does not match the storyboard or leads the player to impossible or undesirable situations.
Most of the games this company produces can be described loosely as race games. Race games are games where the players need to go from an initial square to a final square, performing along the way, a series of moves, gaining or losing score points for each of those moves. Moves can be influenced by player's decisions, drawing of cards, rolling of dices, etc..

Problem

Your task is to check the description of a given game, stating the lowest possible score, or if it can lead to an infinitely high (there's no way the player can win the game), or to an infinitely low score.

Input

The input consists of one game description. The first line of the input contains a positive integer **N** not greater than 300, indicating the number of squares in the game. The second line contains two non-negative integers, **I** and **F**, defining the initial and final squares for this game, where **I** and **F** are lesser than **N**. The third line contains an integer **M**, indicating the number of possible moves of the game. The following **M** lines describe all the possible moves of the game. Each line, describing one possible move, consists of three integers, respectively, the initial square and final square of the move, in the range **[0, N-1]**, and the corresponding score.

Output

The output consists of a single line with an integer, indicating the lowest possible score for the proposed game. If the scores are infinitely high or low then your program should output **`infinity`**.

Sample Input 1

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

Sample Output 1

```
6
```

Sample Input 2

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

Sample Output 2

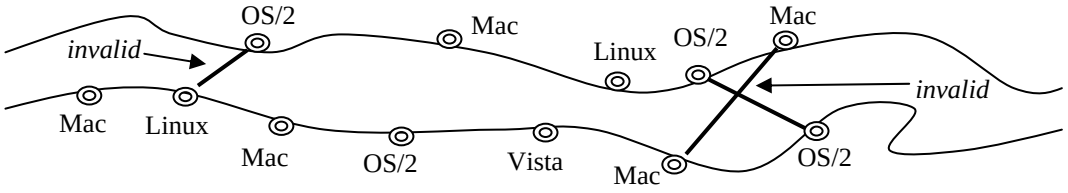
```
infinity
```

SWERC 2007 Problem Set -- Lisbon, 18 November 2007

Problem F - The Bridges of Kölsberg

Background

King Beer has a very hard region to rule, consisting of lots of cities with very sectarian operating system beliefs and high levels of trade. These cities are placed along a river, the Kölsberg, along its Northern and Southern banks. The cities are economically separated from each other, since the river is wide and dangerous.



A section of the Kölsberg showing some invalid bridges

King Beer would like to build some bridges connecting opposite banks of the river. He was strongly advised against making bridges between cities with different operating systems beliefs (those guys really hate each other). So, he is just going to build bridges between cities sharing the same operating system belief (even if the resulting bridges are quite long and strangely shaped). However, it is technical impossible to build bridges that cross other bridges.

The economical value of a bridge is the sum of the trade values the two cities it connects. The King wants to maximize the sum of all possible bridge values while minimizing the number of bridges to build.

Problem

Given two sets of cities, return the maximum possible sum of all bridge values and the smallest number of valid bridges necessary to achieve it.

Input

The first line is an integer with the number of samples. For each sample, the next line has a non-negative integer, not greater than 1,000, indicating the number of cities on the Northern riverbank. Then, on each line, comes the city information with the form

cityname ostype tradevalue

where, separated by empty spaces, there are two strings, *cityname* and *ostype*, with no more than 10 characters each, and *tradevalue* which is a non-negative integer not greater than 10^6. The sequence of lines represents the cities from left to right along the riverbank. Next, there is the same kind of information to describe the Southern riverbank.

Output

For each sample, a line consisting of the maximum possible sum of all bridge values, one empty space, the number of bridges.

Sample Input

```
1
3
mordor Vista 1000000
xanadu Mac 1000
```

```
shangrila OS2 400
4
atlantis Mac 5000
hell Vista 1200
rivendell OS2 100
appleTree Mac 50
```

Sample Output

```
1002250 2
```

Problem G - The Finest Chef

Background

The World Finest Young Chef Competition welcomes young chefs from around the world. Making resources available for them to work is a very difficult job, because we cannot know, beforehand, what kind of equipment they are going to need. Each chef will be aiming to cook his best dish, but this can involve a stove, a fridge, a freezer, a microwave oven, etc. Each dish will only need one of these equipments once, for a limited period of time, but this period will vary, depending on the equipment used. For example, it is possible that one dish can be accomplished either using a fridge (using 30 minutes of the fridge's time) or a freezer (using only 5 minutes). Moreover, each equipment can only be used by one chef, for the duration of the competition, as after being used, it will need cleaning.

Problem

The aim is to find, for each chef, an equipment that will suit their dish, minimizing the sum of the cooking times of all of the dishes in competition.

Input

The input begins with a line containing two positive integers to indicate the number of chefs in the competition, not greater than 250, and the number of facilities available to cook, not greater than 350. The next line contains a single integer, the number of lines to be read next. The following lines describe how long one chef's dish takes to cook in a specific facility, in the following way: one non-negative integer identifying the chef, one non-negative integer identifying the facility and a third positive integer for the cooking time. It is guarantied that there are enough facilities to cook all dishes.

Output

The output is one single line, which contains an integer with the sum of the cooking times for all the dishes in the competition.

Sample input 1

```
4 5
9
0 2 5
0 3 3
1 1 20
1 4 10
2 1 25
2 4 30
3 0 2
3 2 10
3 3 12
```

Sample output 1

```
40
```

Sample input 2

```
3 3
9
```

```
0 0 3
0 1 2
0 2 1
1 0 1
1 1 7
1 2 9
2 0 3
2 1 7
2 2 5
```

Sample output 2

```
8
```

Problem H - IP-TV

Background

A consortium of European Internet providers manages a large backbone network, with direct links (connections) between a large number of European cities. A link between a pair of cities is bidirectional. The transmission of a message in a link has an associated cost. As it is common in the Internet, it is possible to use a (unbounded) sequence of direct links to indirectly transfer data between any pair of cities.

For allowing the broadcast of TV programs using this backbone, it is necessary to continuously send data to all nodes in the network. For helping to minimize costs, it is necessary to select the network links that will be used for transmitting data. The set of selected links must be connected and include all nodes in the network.

For helping the consortium to manage its network, you have been asked to create a program that computes the minimum cost for transmitting data to all cities of the backbone.

Problem

Given a set of network links, compute the minimum transmission cost for reaching all nodes.

Input

The first line of the input contains a positive integer **M**, not greater than 2,000, with the number of cities that have network connections. The second line contains an integer **N** not greater than 50,000, with the number of existing links. Each of the following **N** lines contains the representation of a link. Each line contains two strings and one integer, separated by empty spaces, **B E C**, where **B** and **E** are the city names of the endpoints of the network link, with no more than 8 characters, and **C** is a positive integer, not greater than 30, representing the cost of transmitting in the link.

Output

The output consists of one single line that contains an integer with the minimum transmission cost for sending data to all cities.

Sample Input

```
4
5
lisbon london 6
lisbon paris 5
london paris 1
london berlin 2
paris berlin 10
```

Sample Output

```
8
```

Problem I – Ladies' Choice

Background

Teenagers from the local high school have asked you to help them with the organization of next year's Prom. The idea is to find a suitable date for everyone in the class in a fair and civilized way. So, they have organized a web site where all students, boys and girls, state their preferences among the class members, by ordering all the possible candidates. Your mission is to keep everyone as happy as possible. Assume that there are equal numbers of boys and girls.

Problem

Given a set of preferences, set up the blind dates such that there are no other two people of opposite sex who would both rather have each other than their current partners. Since it was decided that the Prom was Ladies' Choice, we want to produce the best possible choice for the girls.

Input

The input consists of a positive integer **N**, not greater than 1,000, indicating the number of couples in the class. Next, there are **N** lines, each one containing the all the integers from 1 to **N**, ordered according to the girl's preferences. Next, there are **N** lines, each one containing all the integers from 1 to **N**, ordered according to the boy's preferences.

Output

The output consists of a sequence of **N** lines, where the *i-th* line contains the number of the boy assigned to the *i-th* girl (from 1 to **N**).

Sample Input

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

Sample Output

```
1
2
5
3
4
```