Problem A: Ornaments

Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

You just bought yourself a Christmas tree, so now you want to find out how many Christmas ornaments you can hang on it. The tree consists of a trunk, surrounded by many branches. Some of them grow directly out of the trunk, the rest is connected to it indirectly over a series of forkings and other branches.

Ornaments can only be hung on branches that end in a tip. Furthermore, the number of ornaments that can be hung on one of those "outer branches" depends on its own thickness as well as the thicknesses of the branches on the way between it and the trunk. Specifically, an outer branch of thickness t can only carry t ornaments and an inner branch of thickness t can only fork into parts of the tree that collectively don't carry more than t ornaments.

Given all branches and their thicknesses, find out how many ornaments can be hung on the tree.

Input

To describe the branches, the forkings and tips are numerated $1, ..., N$. The input consists of

- one line containing N ($1 \leq N \leq 10^5$) – the total number of forkings and tips
- N lines describing the branches, with the i-th line containing three integers a_i, b_i and t_i ($0 \leq a_i \leq N, 1 \leq b_i \leq N, 1 \leq t_i \leq 10^4$), denoting a branch between the objects with indices a_i and b_i (trunk–tip, trunk–forking, forking–forking or forking–tip, where the trunk has index 0 and a_i is guaranteed to be closer to the trunk than b_i) that has thickness t_i.

Output

Output the maximum number of ornaments you can hang on the outer branches of your tree.

Sample input and output

Input	Output
9 0 1 10 1 2 4 1 3 7 0 4 8 0 5 12 5 6 8 6 7 1 6 8 2 5 9 1	22

Problem B: Gifting

Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

Some of the residents of your town have already made gifts to each other for Christmas, but today the mayor declared that every resident of his town must make precisely as many such gifts as they recieve. Now they need your programming skills to calculate how many gifts are still missing to fulfill the mayors demand, based on a list of those already made.

Input

The input consists of

- one line containing N and M ($2 \leq N \leq 10^5, 1 \leq M \leq 3 \cdot 10^5$) – the number of people and the amount of gifts they already made
- M lines describing the gifts, with the i-th line containing integers x_i and y_i ($1 \leq x_i, y_i \leq N$), meaning that person x_i has made a gift to person y_i.

Output

Output the number of missing gifts. One can receive multiple gifts from the same person.

Sample input and output

Input	Output
3 4 1 2 1 3 2 3 3 1	1

Problem C: Tickets

Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

Due to some bureaucratic difficulties, Santa didn't manage to renew the flight permit for his sleigh in some countries. Of course he didn't want to let this stop him from delivering his presents to these countries, so he decided to travel them by train instead.

Unfortunately, covering long distances by train is very expensive and Santa already spent most of his money on all the things he and his helpers needed to craft the presents. Therefore he asked you to write a program that tells him which train tickets he needs to buy in each of the countries, in order to be able to reach every city in it while spending as little as possible.

Input

The input consists of

- one line containing N and M ($2 \leq N \leq 10^4, 1 \leq M \leq 1000$) – the numbers of cities and train lines, respectively
- M lines describing the train lines, with the i-th line beginning with T_i and C_i ($2 \leq T_i \leq 1000, 1 \leq C_i \leq 1000$) – the number of train stations in the i-th train line and the cost of a ticket between any two neighboring (according to the order in which they are given) cities in that line. Then follow T_i numbers $t_{i,1}, ..., t_{i,T_i}$ ($1 \leq t_{i,j} \leq N$), the indices of the train stations that it covers.

As indicated above, any individual ticket is just valid for a connection between two cities that are directly connected (without intermediate stops) by one of the train lines, but once one has bought such a ticket, it can be used to travel between those cities in both directions arbitrarily often for the rest of the year (which suffices for Santa).

Output

Output the minimal amount of money with which Santa can buy tickets that allow him to travel (not necessarily directly) between all cities in that country. It is guaranteed that every city can be reached from any other one by the given connections.

Sample input and output

Input	**Output**
3 2 2 150 1 2 3 100 1 2 3	200
5 2 4 350 1 2 3 5 3 150 2 4 5	1000

Problem D: Flight Path

Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

Since Santa has to be extremely efficient to deliver billions of presents in a single night, every year he is very concerned about finding the ideal flight paths between many pairs of cities. This is an especially difficult task because the travel time of each route highly depends on the local wind directions in the overflown areas, which of course vary from year to year and can therefore only be assumed to be constant for the night.

Since he recently heard about your tremendous programming skills, he kindly asked you to provide him with a long-term solution for his problem: A program that, for any region given by its wind directions and the coordinates of the cities located there, calculates the cost of the path Santa should take through it (given that information, he will be able figure out the actual path by himself). The order in which Santa wants to visit the given cities depends on other factors also, so he will decide that part for you.

Input

Winds in a region are described by a rectangular grid, where each cell contains the current wind direction in that cell. Each city covers precisely one of those cells. To understand the calculation of a route's cost, consider not traveling from cell to cell, but rather traveling *trough* the cells and *between* the points halving the shared borders of adjacent cells. The cost of flying straight through a cell is 1 when traveling with the wind, 2 when traveling sideways to it and 5 when traveling directly against it. The cost of taking a 90 degree turn in a cell, e.g. starting at the left side of it and going to the top, equals the maximum of the costs you would have for traveling straight through it in either of the directions you coincide with, so left-to-right and bottom-to-top in the example (this is due to the fact that taking a turn will slow Santa down). To reach a city you just have to reach any of the sides of the cell it covers and when starting to travel from a city you just visited, you may also start from any side of that city's cell, not necessarily the side you arrived on.

The input consists of

- one line containing N, M and C ($1 \leq N, M, C \leq 50$) – the amounts of rows and columns in the grid as well as the number of cities in the region
- one line containing s and e ($0 \leq s, e < N$), describing that Santa starts at the left side of cell $(s, 1)$ and wants to end up on the right side of cell (e, M) after visiting all cities
- N lines, each containing M of the symbols $<$, $>$, $\wedge$, $\vee$ (the latter two are given as circumflex and small letter v in the actual input), describing the wind directions in the respective cells (i.e. $> \mathrel{\widehat{=}}$ left-to-right-wind etc).
- C lines describing the cities in the order in which Santa wants to visit them, denoting each city by the (matrix-style) coordinates (i, j) of the cell it is in ($0 \leq i < N, 0 \leq j < M$).

Note that a cell with a city in it can be overflown without visiting that city.

Output

Output a single number, the minimum cost Santa will have for starting at the described starting point, visiting all cities in the given order and ending up at the described ending point.

Sample input and output

Input	**Output**
1 1 1 0 0 v 0 0	0
3 4 2 0 2 < v v < ^ < v v ^ ^ < < 1 1 0 3	20

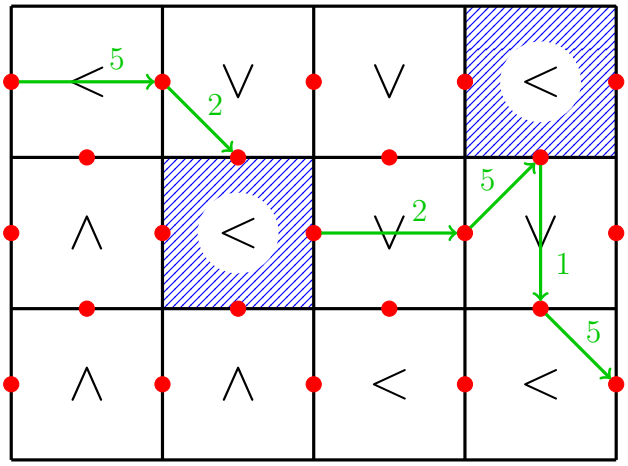

Illustration of the second sample. Cells with cities in them are marked blue, the points between which Santa needs to travel marked in red and one of the optimal paths (including the costs of each route segment) is indicated in green.

Problem E: Regulations

Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

Due to new Christmas regulations in Germany, Santa is only allowed to visit the homes of children living in each German town in a certain order. Furthermore, he is only allowed to visit a child whose name begins with the same letter the name of the last child he visited ends with. In order to abide by these regulations, Santa will probably have to leave out some children. Given the names of the children in a town and the order he needs to follow, you are to find the maximum amount of children he can visit.

Input

The input consists of

- one line containing n ($1 \leq n \leq 10^4$) – the number of children
- n lines containing the names of the children (i.e. strings of length ≤ 15) in the order Santa needs to follow. If a name appears multiple times, it refers to different children sharing that name.

Output

Output a single number, the maximum amount of children Santa can visit when abiding the regulations.

Sample input and output

Input	Output
4 Marie Elisabeth Michael Hans	3

Problem F: Presents

Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

Whenever Santa stops his sleigh to climb down a chimney, he has to pack all the gifts he wants to bring along into two bags. But sometimes he has so many presents for one house that he can't fit them all in. In such a case he has to leave some of them on the sleigh and save them for the next year, since he has to operate extremely fast to deliver everything in time. Of course he wants to bring as much joy as possible, so he asked you to write a program, that – given the sizes of the two bags, the sizes of the individual presents and how happy each of them would make its recipient – calculates the maximum amount of joy he can bring to that house (he will then be able to figure out on his own, which presents he needs to take to achieve this).

Input

The input consists of

- one line containing N, S_1 and S_2 ($1 \leq N, S_1, S_2 \leq 500$) – the number of presents and the sizes of the bags, respectively
- N lines describing the presents, with the i-th line containing integers s_i and j_i ($1 \leq s \leq 500$, $1 \leq j \leq 1000$), denoting a present of size s_i that brings j_i joy.

Output

Output a single number, the maximum amount of joy Santa can bring with the presents as described above.

Sample input and output

Input	Output
4 9 7 3 20 6 40 4 20 3 20	100