Problem A

BUSES

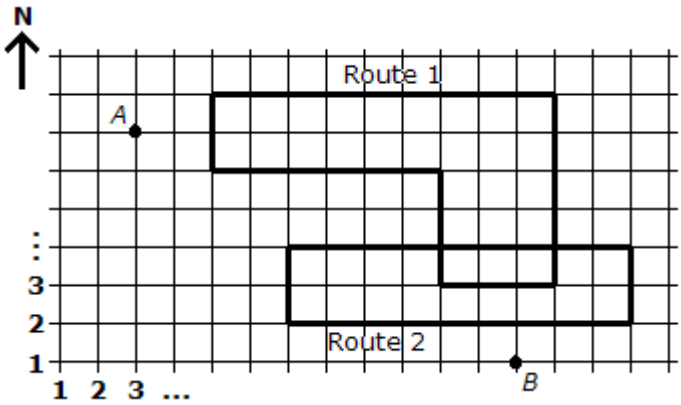
Portsaid is a north-eastern Egyptian city that lies near the Suez Canal. All streets in Portsaid are perfect straight lines oriented either north-south or east-west. Note that an intersection is a place where two streets cross each others. The public transport system of Portsaid consists of **R** bus routes. A route is a predefined cycle inside the city that is followed by a bus stopping at every intersection on that route. Each route has exactly one bus assigned to it that follows its path. Every bus has a fee **fi**, once paid you can take the bus as long as you want, but once you get off, you'll have to pay the bus fee again for another ride.

Since not every intersection in the city lies in the path of a route, sometimes you have to walk to be able to move from some intersection to another. When walking from one intersection to another, you can only use the streets.

Today you are in Portsaid for vacation and you want to go from intersection A to intersection B. Since you are here for vacation, you decided not to walk more than **D** blocks (A block is a distance that separates two consecutive intersections along a street). Also, you want to spend as little money as you can.

TASK

You are to write a program that is given the map of the bus routes and their fees, computes the minimum amount of money you have to spend to go from **A** to **B** walking at most **D** blocks.



To describe the city we will number North-South streets starting from the west and moving east, and East-West streets starting from the South and moving North as shown in figure. Intersections will be described as a pair (x, y) where x represents the North-South (vertical) street and y represents the East-West (horizontal) street.

INPUT

- 1st line consists of an integer **D** (0 <= **D** <= 300).
- 2nd line consists of two integers separated by single space that represent intersection **A**.
- 3rd line consists of two integers separated by single space that represent intersection **B**.
- Both coordinates of **A** and **B** range from 1 to 100,000,000 inclusive.
- **A** and **B** will never represent the same intersection.

- 4th consists of an integer **R** (1 <= **R** <= 100), the number of bus routes in the city.
- Each of the next **R** lines will describe a bus route; each route will be described as a sequence of integers separated by single spaces as follows:
  - The first integer on the line represents **Ni** (4 <= **Ni** <= 50), the number of intersections used to describe this route.
  - The second integer on the line represents $\mathbf{f_i}$, the fee associated with that route. (0 <= $\mathbf{f_i}$ <= 1,000,000).
  - Then follows **Ni** pairs of integers, each represents an intersection. The bus starts moving from the first intersection and drives in a straight line to the second; it turns 90 degrees and drives in a straight line to the third, etc. It keeps moving that way until it reaches the last intersection on the description; from there it turns 90 degrees and drives to the first one to start again. Note, that the ith route has **Ni** line segments which never overlap and never intersect except in the intersection point between any 2 consecutive segments. All coordinates will range from 1 to 100,000,000 inclusive.

OUTPUT

The output must be one integer representing the minimum amount of money you need to spend to get from **A** to **B** walking at most **D** blocks. If there is no possible way to achieve that, your program should output **–1**.

GRADING

In some cases worth 30 points:

- **D** does not exceed 100.
- **A**, **B** and the polygons' coordinates do not exceed 100.
- **R** does not exceed 25.
- **Ni** does not exceed 10.
- **fi** does not exceed 10.
- Any line segment in any polygon does not have more than 40 lattice points including the starting and ending points of that line segment.

DETAILED FEEDBACK

During the contest, your submissions for this task will be evaluated on part of the official test data showing you a summary of the results.

EXAMPLE

Sample Input 1	Sample Output 1
4 3 7 13 1 2 6 2 14 8 5 8 5 6 11 6 11 3 14 3 4 5 16 4 7 4 7 2 16 2	2

This sample input corresponds to the shown figure. Note that the best way here is to move for 2 blocks, then use the first bus route, then walk for 2 blocks to reach B.

Sample Input 2	Sample Output 2
2 1 5 10 7 3 4 10 1 4 5 4 5 6 1 6 4 10 5 5 5 7 7 7 7 5 4 20 9 5 9 1 7 1 7 5	-1

Problem C

ISLANDS

You are visiting a park which has ***N*** islands. From each island ***i***, exactly one bridge was constructed. The length of that bridge is denoted by L_i. The total number of bridges in the park is ***N***. Although each bridge was built from one island to another, now every bridge can be traversed in both directions. Also, for each pair of islands, there is a unique ferry that travels back and forth between them.

Since you like walking better than riding ferries, you want to maximize the sum of the lengths of the bridges you cross, subject to the constraints below.

- You can start your visit at an island of your choice.
- You may not visit any island more than once.
- At any time you may move from your current island ***S*** to another island ***D*** that you have **not** visited before. You can go from ***S*** to ***D*** either by:
  - Walking: Only possible if there is a bridge between the two islands. With this option the length of the bridge is added to the total distance you have walked, or
  - Ferry: You can choose this option only if ***D*** is not reachable from ***S*** using any combination of bridges and/or previously used ferries. (When checking whether it is reachable or not, you consider all paths, including paths passing through islands that you have already visited.)

Note that you do not have to visit all the islands, and it may be impossible to cross all the bridges.

TASK

Write a program that, given the ***N*** bridges along with their lengths, computes the maximum distance you can walk over the bridges obeying the rules described above.

CONSTRAINTS

$2 <= N <= 1{,}000{,}000$	The number of islands in the park.
$1 <= L_i <= 100{,}000{,}000$	The length of bridge ***i***.

INPUT

Your program must read from the standard input the following data:

- Line 1 contains the integer ***N***, the number of islands in the park. Islands are numbered from 1 to ***N***, inclusive.
- Each of the next ***N*** lines describes a bridge. The i^{th} of these lines describes the bridge constructed from island ***i*** using two integers separated by a single space. The first integer represents the island at the other endpoint of the bridge, the second integer represents the length L_i of the bridge. You may assume that for each bridge, its endpoints are always on two different islands.

OUTPUT

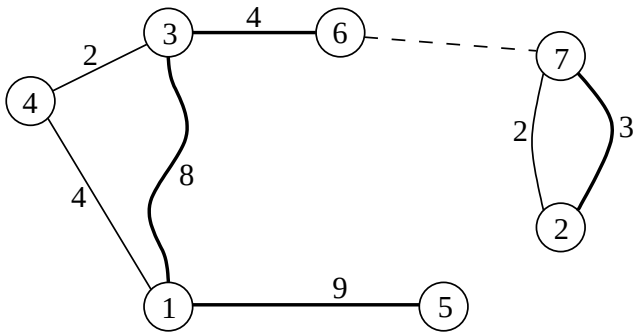
Your program must write to the standard output a single line containing one integer, the maximum possible walking distance.

NOTE 1: For some of the test cases the answer will not fit in a 32-bit integer, you might need int64 in Pascal or long long in C/C++ to score full points on this problem.
NOTE 2: When running Pascal programs in the contest environment, it is significantly slower to read in 64-bit data types than 32-bit data types from standard input even when the values being read in fit in 32 bits. We recommend that you read the input data into 32-bit data types.

GRADING
For some cases worth 40 points, ***N*** will not exceed 4,000.

EXAMPLE

Sample input	Sample output
7 3 8 7 2 4 2 1 4 1 9 3 4 2 3	24



The ***N=7*** bridges in the sample are (1-3), (2-7), (3-4), (4-1), (5-1), (6-3) and (7-2). Note that there are two different bridges connecting islands 2 and 7.
One way that you can achieve maximum walking distance follows:

- Start on island 5.
- Walk the bridge of length 9 to reach island 1.
- Walk the bridge of length 8 to reach island 3.
- Walk the bridge of length 4 to reach island 6.
- Take the ferry from island 6 to island 7.
- Walk the bridge of length 3 to reach island 2.

By the end you are on island 2 and your total walking distance is 9+8+4+3 = 24.
The only island that was not visited is island 4. Note that at the end of the trip described above you can not visit this island any more. More precisely:

- You are not able to visit it by walking, because there is no bridge connecting island 2 (where you currently stand) and island 4.
- You are not able to visit it using a ferry, because island 4 is reachable from island 2, where you currently stand. A way to reach it: use the bridge (2-7), then use a ferry you already used to get from island 7 to island 6, then the bridge (6-3), and finally the bridge (3-4).

Problem D

FISH

It was told by Scheherazade that far away, in the middle of the desert, there is a lake. Originally this lake had ***F*** fish in it. ***K*** different kinds of gemstones were chosen among the most valuable on Earth, and to each of the ***F*** fish exactly one gem was given for it to swallow. Note, that since ***K*** might be less than ***F***, two or more fish might swallow gems of the same kind.

As time went by, some fish ate some of the other fish. One fish can eat another if and only if it is at least twice as long (fish ***A*** can eat fish ***B*** if and only if $L_A >= 2 * L_B$). There is no rule as to when a fish decides to eat. One fish might decide to eat several smaller fish one after another, while some fish may decide not to eat any fish, even if they can. When a fish eats a smaller one, its length doesn't change, but the gems in the stomach of the smaller fish end up undamaged in the stomach of the larger fish.

Scheherazade has said that if you are able to find the lake, you will be allowed to take out one fish and keep all the gems in its stomach for yourself. You are willing to try your luck, but before you head out on the long journey, you want to know how many different combinations of gems you could obtain by catching a single fish.

TASK

Write a program that given the length of each fish and the kind of gemstone originally swallowed by each fish, finds **the number of different combinations of gems that can end up in the stomach of any fish, modulo some given integer *M*.** A combination is defined only by the number of gems from each of the ***K*** kinds. There is no notion of order between gems, and any two gems of the same kind are indistinguishable.

CONSTRAINTS

1 <= ***F*** <= 500,000	The original number of fish in the lake.
1 <= ***K*** <= ***F***	The number of different gemstone kinds.
2 <=***M*** <= 30,000	
1 <= L_X <= 1,000,000,000	The length of fish *X*.

INPUT

Your program must read from the standard input the following data:

- Line 1 contains the integer ***F***, the original number of fish in the lake.
- Line 2 contains the integer ***K***, the number of kinds of gemstones.

The kinds of gemstones are represented by integers 1 to ***K***, inclusive.

- Line 3 contains the integer ***M***.
- Each of the following ***F*** lines describes one fish using 2 integers separated by a single space: the length of the fish followed by the kind of gemstone originally swallowed by that fish.

NOTE: For all test cases used for evaluation, it is guaranteed that there is at least one gemstone from each of the ***K*** kinds.

OUTPUT

Your program must write to the standard output a single line containing one integer between 0 and ***M-1*** (inclusive): the number of different possible combinations of gemstones **modulo *M***.
Note that for solving the task, the value of ***M*** has no importance other than simplifying computations.

GRADING

For a number of tests, worth a total of 70 points, ***K*** will not exceed 7,000.
Also, for some of these tests, worth a total of 25 points, ***K*** will not exceed 20.

DETAILED FEEDBACK

During the contest, your submissions for this task will be evaluated on some of the official test data showing you a summary of the results.

EXAMPLE

Sample Input	Sample Output
5 3 7 2 2 5 1 8 3 4 1 2 3	4

There are 11 possible combinations so you should output 11 modulo 7 which is 4.

The possible combinations are: [1] [1,2] [1,2,3] [1,2,3,3] [1,3] [1,3,3] [2] [2,3] [2,3,3] [3] and [3,3].
(For each combination, we list the gems it contains. For example, [2,3,3] is a combination that consists of one gem of kind 2, and two gems of kind 3.)
These combinations can be achieved in the following ways:

- [1]: It is possible that you catch the second (or the fourth) fish before it eats any other fish.
- [1,2]: If the second fish eats the first fish, then it would have a gemstone of kind 1 (the one it originally swallowed) and a gemstone of kind 2 (from the stomach of the first fish).
- [1,2,3]: One possible way of reaching this combination: the fourth fish eats the first fish, and then the third fish eats the fourth fish. If you now catch the third fish, it will have one gemstone of each kind in its stomach.
- [1,2,3,3]: Fourth eats first, third eats fourth, third eats fifth, you catch the third one.
- [1,3]: Third eats fourth, you catch it.
- [1,3,3]: Third eats fifth, third eats fourth, you catch it.
- [2]: You catch the first fish.
- [2,3]: Third eats first, you catch it.
- [2,3,3]: Third eats first, third eats fifth, you catch it.
- [3]: You catch the third fish.
- [3,3]: Third eats fifth, you catch it.

Problem E

PYRAMID BASE

You have been asked to find the largest affordable location for constructing a new pyramid. In order to help you decide, you have been provided with a survey of the available land which has been conveniently divided into an ***M*** by ***N*** grid of square cells. The base of the pyramid must be a square with sides parallel to those of the grid.

The survey has identified a set of ***P*** possibly overlapping obstacles, which are described as rectangles in the grid with sides parallel to those of the grid. In order to build the pyramid, all the cells covered by its base must be cleared of any obstacles. Removing the i^{th} obstacle has a cost C_i. Whenever an obstacle is removed, it must be removed completely, that is, you cannot remove only part of an obstacle. Also, please note that removing an obstacle does not affect any other obstacles that overlap it.

TASK

Write a program that, given the dimensions ***M*** and ***N*** of the survey, the description of the ***P*** obstacles, the cost of removing each of the obstacles, and the budget ***B*** you have, finds the maximum possible side length of the base of the pyramid such that the total cost of removing obstacles does not exceed ***B***.

CONSTRAINTS AND GRADING

Your program will be graded on three disjoint sets of tests. For all of them, the following constraints apply:

$1 \le M, N \le 1{,}000{,}000$	The dimensions of the grid.
$1 \le C_i \le 7{,}000$	The cost of removing the i^{th} obstacle.
$1 \le X_{i1} \le X_{i2} \le M$	X coordinates of the leftmost and the rightmost cells of the i^{th} obstacle.
$1 \le Y_{i1} \le Y_{i2} \le N$	Y coordinates of the bottommost and the topmost cells of the i^{th} obstacle.

<u>In the first set of tests worth 35 points:</u>

$B = 0$	The budget you have. (You cannot remove any obstacles.)
$1 \le P \le 1{,}000$	The number of obstacles in the grid.

<u>In the second set of tests worth 35 points:</u>

$0 < B \le 2{,}000{,}000{,}000$	The budget you have.
$1 \le P \le 30{,}000$	The number of obstacles in the grid.

<u>In the third set of tests worth 30 points:</u>

$B = 0$	The budget you have. (You cannot remove any obstacles.)
$1 \le P \le 400{,}000$	The number of obstacles in the grid.

INPUT

Your program must read from the standard input the following data:

- Line 1 contains two integers separated by a single space that represent ***M*** and ***N*** respectively.
- Line 2 contains the integer ***B***, the maximum cost you can afford (i.e., your budget).
- Line 3 contains the integer ***P***, the number of obstacles found in the survey.

- Each of the next ***P*** lines describes an obstacle. The i^{th} of these lines describes the i^{th} obstacle. Each line consists of 5 integers: X_{i1}, Y_{i1}, X_{i2}, Y_{i2}, and C_i separated by single spaces. They represent respectively the coordinates of the bottommost leftmost cell of the obstacle, the coordinates of the topmost rightmost cell of the obstacle, and the cost of removing the obstacle. The bottommost leftmost cell on the grid has coordinates (1, 1) and the topmost rightmost cell has coordinates (***M***, ***N***).

OUTPUT

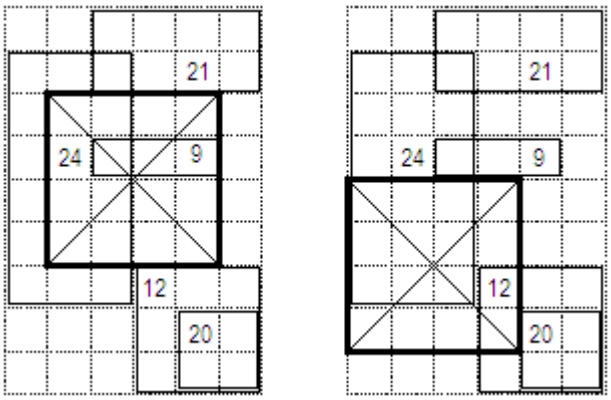
Your program must write to the standard output a single line containing one integer, the maximum possible side length of the base of the pyramid that can be prepared. If it is not possible to build any pyramid, your program should output the number 0.

DETAILED FEEDBACK

During the contest, your submissions for this task will be evaluated on some of the official test data showing you a summary of the results.

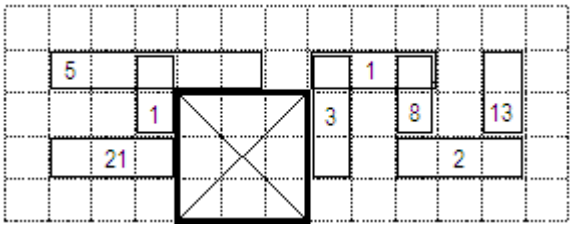
EXAMPLE

Sample input 1	Sample output 1
6 9 42 5 4 1 6 3 12 3 6 5 6 9 1 3 3 8 24 3 8 6 9 21 5 1 6 2 20	4



The figure shows two possible locations for the pyramid's base, both having a side of length 4.

Sample input 2	Sample output 2
13 5 0 8 8 4 10 4 1 4 3 4 4 1 10 2 12 2 2 8 2 8 4 3 2 4 6 4 5 10 3 10 4 8 12 3 12 4 13 2 2 4 2 21	3



The figure shows the only possible location for the pyramid's base having a side of length 3.