A. Square and Rectangles

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given n rectangles. The corners of rectangles have integer coordinates and their edges are parallel to the Ox and Oy axes. The rectangles may touch each other, but they do not overlap (that is, there are no points that belong to the interior of more than one rectangle).

Your task is to determine if the rectangles form a square. In other words, determine if the set of points inside or on the border of at least one rectangle is precisely equal to the set of points inside or on the border of some square.

Input

The first line contains a single integer n $(1 \le n \le 5)$. Next n lines contain four integers each, describing a single rectangle: x_1, y_1, x_2, y_2 $(0 \le x_1 < x_2 \le 31400, 0 \le y_1 < y_2 \le 31400)$ — x_1 and x_2 are x-coordinates of the left and right edges of the rectangle, and y_1 and y_2 are y-coordinates of the bottom and top edges of the rectangle.

No two rectangles overlap (that is, there are no points that belong to the interior of more than one rectangle).

Output

In a single line print "`YES`", if the given rectangles form a square, or "`NO`" otherwise.

Sample test(s)

input

```
5
0 0 2 3
0 3 3 5
2 0 5 2
3 2 5 5
2 2 3 3
```

output

```
YES
```

input

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

output

```
NO
```

B. Stadium and Games

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Daniel is organizing a football tournament. He has come up with the following tournament format:

1. In the first several (possibly zero) stages, while the number of teams is even, they split in pairs and play one game for each pair. At each stage the loser of each pair is eliminated (there are no draws). Such stages are held while the number of teams is even.
2. Eventually there will be an odd number of teams remaining. If there is one team remaining, it will be declared the winner, and the tournament ends. Otherwise each of the remaining teams will play with each other remaining team once in round robin tournament (if there are x teams, there will be $\frac{x \cdot (x-1)}{2}$ games), and the tournament ends.

For example, if there were 20 teams initially, they would begin by playing 10 games. So, 10 teams would be eliminated, and the remaining 10 would play 5 games. Then the remaining 5 teams would play 10 games in a round robin tournament. In total there would be 10+5+10=25 games.

Daniel has already booked the stadium for n games. Help him to determine how many teams he should invite so that the tournament needs **exactly** n games. You should print all possible numbers of teams that will yield exactly n games in ascending order, or `-1` if there are no such numbers.

Input

The first line contains a single integer n ($1 \le n \le 10^{18}$), the number of games that should be played.

Please, do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Output

Print all possible numbers of invited teams in ascending order, one per line. If exactly n games cannot be played, output one number: `-1`.

Sample test(s)

input
```
3
```
output
```
3
4
```

input
```
25
```
output
```
20
```

input
```
2
```
output
```
-1
```

C. Monsters and Diamonds

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Piegirl has found a monster and a book about monsters and pies. When she is reading the book, she found out that there are n types of monsters, each with an ID between 1 and n. If you feed a pie to a monster, the monster will split into some number of monsters (possibly zero), and at least one colorful diamond. Monsters may be able to split in multiple ways.

At the begining Piegirl has exactly one monster. She begins by feeding the monster a pie. She continues feeding pies to monsters until no more monsters are left. Then she collects all the diamonds that were created.

You will be given a list of split rules describing the way in which the various monsters can split. Every monster can split in at least one way, and if a monster can split in multiple ways then each time when it splits Piegirl can choose the way it splits.

For each monster, determine the smallest and the largest number of diamonds Piegirl can possibly collect, if initially she has a single instance of that monster. Piegirl has an unlimited supply of pies.

Input

The first line contains two integers: m and n ($1 \le m, n \le 10^5$), the number of possible splits and the number of different monster types. Each of the following m lines contains a split rule. Each split rule starts with an integer (a monster ID) m_i ($1 \le m_i \le n$), and a positive integer l_i indicating the number of monsters and diamonds the current monster can split into. This is followed by l_i integers, with positive integers representing a monster ID and `-1` representing a diamond.

Each monster will have at least one split rule. Each split rule will have at least one diamond. The sum of l_i across all split rules will be at most 10^5.

Output

For each monster, in order of their IDs, print a line with two integers: the smallest and the largest number of diamonds that can possibly be collected by starting with that monster. If Piegirl cannot possibly end up in a state without monsters, print `-1` for both smallest and the largest value. If she can collect an arbitrarily large number of diamonds, print `-2` as the largest number of diamonds.

If any number in output exceeds 314000000 (but is finite), print 314000000 instead of that number.

Sample test(s)

input

```
6 4
1 3 -1 1 -1
1 2 -1 -1
2 3 -1 3 -1
2 3 -1 -1 -1
3 2 -1 -1
4 2 4 -1
```

output

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

input

```
3 2
1 2 1 -1
2 2 -1 -1
2 3 2 1 -1
```

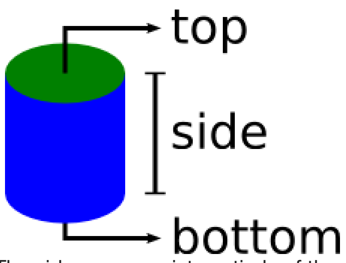
output

```
-1 -1
2 2
```

D. Reclamation

time limit per test: 5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

In a far away land, there exists a planet shaped like a cylinder. There are three regions in this planet: top, bottom, and side as shown in the following picture.



Both the top and the bottom areas consist of big cities. The side area consists entirely of the sea.

One day, a city decides that it has too little space and would like to reclamate some of the side area into land. The side area can be represented by a grid with r rows and c columns — each cell represents a rectangular area in the side area. The rows are numbered 1 through r from top to bottom, while the columns are numbered 1 through c from left to right. Two cells are adjacent if they share a side. In addition, two cells located on the same row — one in the leftmost column, and the other in the rightmost column — are also adjacent.

Initially, all of the cells are occupied by the sea. The plan is to turn some of those cells into land one by one in a particular order that will be given to you.

However, the sea on the side area is also used as a major trade route. More formally, it is not allowed to reclamate the sea cells into land in such way that there does not exist a sequence of cells with the following property:

- All cells in the sequence are occupied by the sea (i.e., they are not reclamated).
- The first cell in the sequence is in the top row.
- The last cell in the sequence is in the bottom row.
- Consecutive cells in the sequence are adjacent.

Thus, the plan is revised. Each time a cell is going to be turned from sea to land, the city first needs to check whether or not it would violate the above condition by doing that. If it would, then the cell is not turned into land and the plan proceeds into the next cell. Otherwise, the cell is turned into land.

Your job is to simulate this and output the number of cells that were successfully turned into land.

Input

The first line consists of three integers r, c, and n ($1 \le r, c \le 3000$, $1 \le n \le 3 \cdot 10^5$). Then, n lines follow, describing the cells in the order you will reclamate them. Each line will consists of two integers: r_i and c_i ($1 \le r_i \le r$, $1 \le c_i \le c$), which represents the cell located at row r_i and column c_i. All of the lines describing the cells will be distinct.

Output

You should output a single number representing the number of cells that were successfully turned to land.

Sample test(s)

input

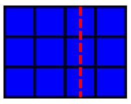
```
3 4 9
2 2
3 2
2 3
3 4
3 1
1 3
2 1
1 1
1 4
```

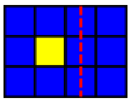
output

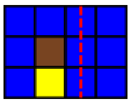
```
6
```

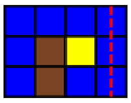
Note

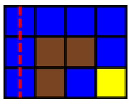
The pictures below show the sequence of reclamations that are performed in the example input. Blue cells represent the cells occupied by sea, while other colored cells represent land. The latest cell that are reclamated is colored either yellow or red, depending on whether the addition violates the condition in the statement. The dashed red line represents a possible trade route, if it exists.

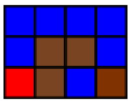




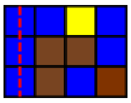


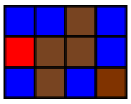




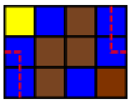


No route exists, so this reclamation is not performed.

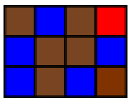




No route exists, skipped.

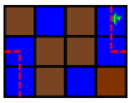


Remember that the leftmost and rightmost cells in the same row are adjacent.



No route exists, skipped.

Hence the result is:



There are 6 successful reclamation and 3 failed ones.

E. The Red Button

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Piegirl found the red button. You have one last chance to change the inevitable end.

The circuit under the button consists of n nodes, numbered from 0 to n - 1. In order to deactivate the button, the n nodes must be disarmed in a particular order. Node 0 must be disarmed first. After disarming node i, the next node to be disarmed must be either node $(2 \cdot i)$ modulo n or node $(2 \cdot i) + 1$ modulo n. The last node to be disarmed must be node 0. Node 0 must be disarmed twice, but all other nodes must be disarmed exactly once.

Your task is to find any such order and print it. If there is no such order, print `-1`.

Input

Input consists of a single integer n ($2 \le n \le 10^5$).

Output

Print an order in which you can to disarm all nodes. If it is impossible, print `-1` instead. If there are multiple orders, print any one of them.

Sample test(s)

input
```
2
```
output
```
0 1 0
```

input
```
3
```
output
```
-1
```

input
```
4
```
output
```
0 1 3 2 0
```

input
```
16
```
output
```
0 1 2 4 9 3 6 13 10 5 11 7 15 14 12 8 0
```