Problem A. New Home

Time limit:	5 seconds
Memory limit:	1024 megabytes

Wu-Fu Street is an incredibly straight street that can be described as a one-dimensional number line, and each building's location on the street can be represented with just one number. Xiao-Ming the Time Traveler knows that there are n stores of k store-types that had opened, has opened, or will open on the street. The i-th store can be described with four integers: x_i, t_i, a_i, b_i, representing the store's location, the store's type, the year when it starts its business, and the year when it is closed.

Xiao-Ming the Time Traveler wants to choose a certain year and a certain location on Wu-Fu Street to live in. He has narrowed down his preference list to q location-year pairs. The i-th pair can be described with two integers: l_i, y_i, representing the location and the year of the pair. Now he wants to evaluate the life quality of these pairs. He defines the inconvenience index of a location-year pair to be the inaccessibility of the most inaccessible store-type of that pair. The inaccessibility of a location-year pair to store-type t is defined as the distance from the location to the nearest type-t store that is open in the year. We say the i-th store is open in the year y if $a_i \le y \le b_i$. Note that in some years, Wu-Fu Street may not have all the k store-types on it. In that case, the inconvenience index is defined as -1.

Your task is to help Xiao-Ming find out the inconvenience index of each location-year pair.

Input

The first line of input contains integer numbers n, k, and q: number of stores, number of types and number of queries ($1 \le n, q \le 3 \cdot 10^5$, $1 \le k \le n$).

Next n lines contain descriptions of stores. Each description is four integers: x_i, t_i, a_i, and b_i ($1 \le x_i, a_i, b_i \le 10^8$, $1 \le t_i \le k$, $a_i \le b_i$).

Next q lines contain the queries. Each query is two integers: l_i, and y_i ($1 \le l_i, y_i \le 10^8$).

Output

Output q integers: for each query output its the inconvenience index.

Scoring

Subtask 1 (points: 5)

$n, q \le 400$

Subtask 2 (points: 7)

$n, q \le 6 \cdot 10^4$, $k \le 400$

Subtask 3 (points: 10)

$n, q \le 3 \cdot 10^5$, $a_i = 1$, $b_i = 10^8$ for all stores.

Subtask 4 (points: 23)

$n, q \le 3 \cdot 10^5$, $a_i = 1$ for all stores.

Subtask 5 (points: 35)

$n, q \le 6 \cdot 10^4$

Subtask 6 (points: 20)

$n, q \le 3 \cdot 10^5$

Examples

input	output
4 2 4 3 1 1 10 9 2 2 4 7 2 5 7 4 1 8 10 5 3 5 6 5 9 1 10	4 2 -1 -1
2 1 3 1 1 1 4 1 1 2 6 1 3 1 5 1 7	0 0 -1
1 1 1 100000000 1 1 1 1 1	99999999

Note

In the first example there are four stores, two types, and four queries.

- First query: Xiao-Ming lives in location 5 in year 3. In this year, stores 1 and 2 are open, distance to store 1 is 2, distance to store 2 is 4. Maximum is 4.
- Second query: Xiao-Ming lives in location 5 in year 6. In this year, stores 1 and 3 are open, distance to store 1 is 2, distance to store 3 is 2. Maximum is 2.
- Third query: Xiao-Ming lives in location 5 in year 9. In this year, stores 1 and 4 are open, they both have type 1, so there is no store of type 2, inconvenience index is -1.
- Same situation in fourth query.

In the second example there are two stores, one type, and three queries. Both stores have location 1, and in all queries Xiao-Ming lives at location 1. In first two queries at least one of stores is open, so answer is 0, in third query both stores are closed, so answer is -1.

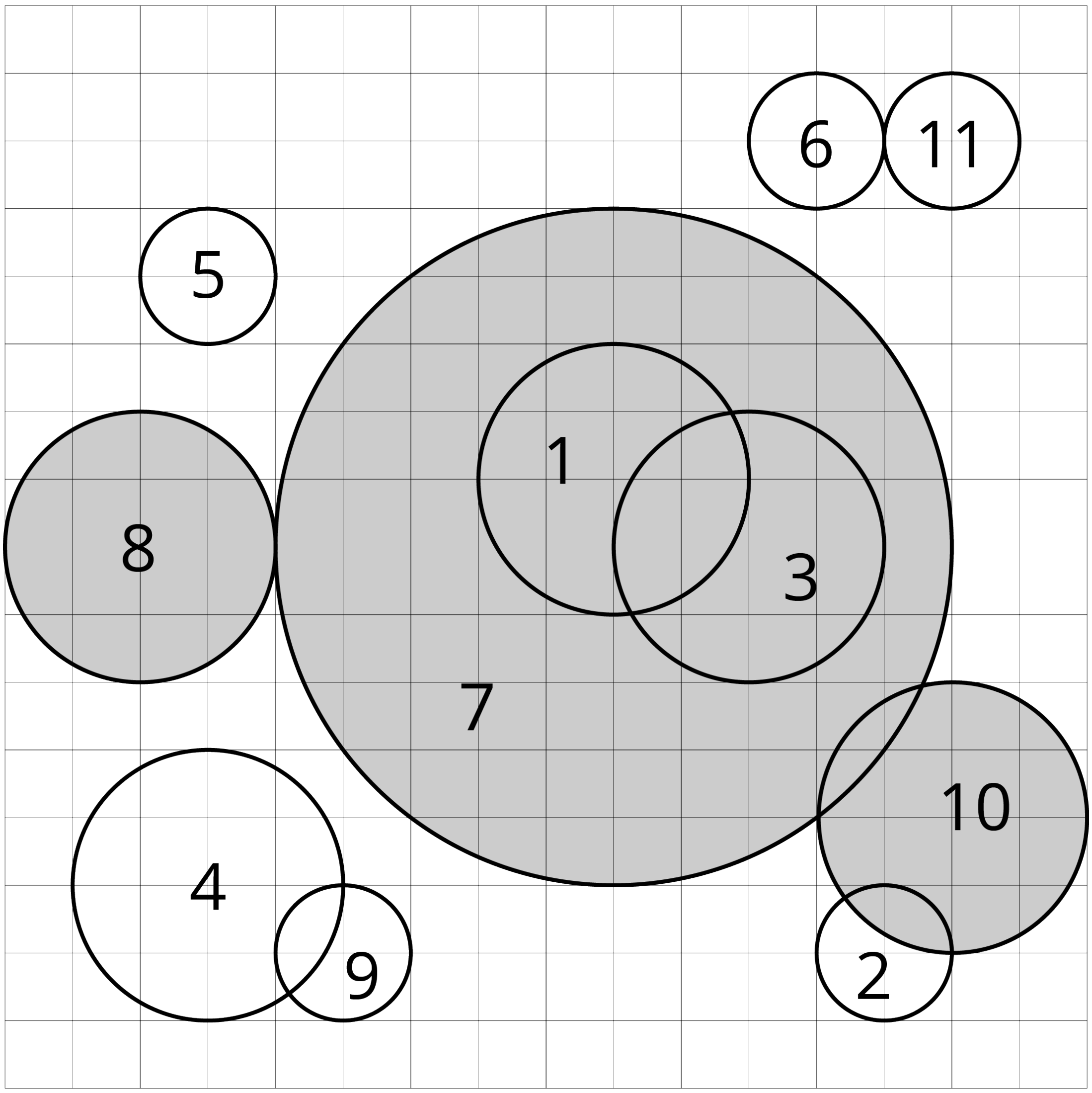
In the third example there is one store and one query. Distance between locations is 99999999.

Problem B. Circle selection

Time limit:	3 seconds
Memory limit:	1024 megabytes

Given n circles $c_1, c_2, \ldots, c_n$ on a flat Cartesian plane. We attempt to do the following:

1. Find the circle c_i with the largest radius. If there are multiple candidates all having the same (largest) radius, choose the one with the smallest index. (i.e. minimize i).
2. Remove c_i and all the circles intersecting with c_i. Two circles intersect if there exists a point included by both circles. A point is included by a circle if it is located in the circle or on the border of the circle.
3. Repeat 1 and 2 until there is no circle left.



We say c_i is eliminated by c_j if c_j is the chosen circle in the iteration where c_i is removed. For each circle, find out the circle by which it is eliminated.

Input

The first line contains an integer n, denoting the number of circles ($1 \le n \le 3 \cdot 10^5$). Each of the next n lines contains three integers x_i, y_i, r_i, representing the x-coordinate, the y-coordinate, and the radius of the circle c_i ($-10^9 \le x_i, y_i \le 10^9$, $1 \le r_i \le 10^9$).

Output

Output n integers $a_1, a_2, \ldots, a_n$ in the first line, where a_i means that c_i is eliminated by c_{a_i}.

Scoring

Subtask 1 (points: 7)

$n \le 5000$

Subtask 2 (points: 12)

$n \le 3 \cdot 10^5$, $y_i = 0$ for all circles

Subtask 3 (points: 15)

$n \le 3 \cdot 10^5$, every circle intersects with at most 1 other circle

Subtask 4 (points: 23)

$n \le 3 \cdot 10^5$, all circles have the same radius.

Subtask 5 (points: 30)

$n \le 10^5$

Subtask 6 (points: 13)

$n \le 3 \cdot 10^5$

Example

input	output
11 9 9 2 13 2 1 11 8 2 3 3 2 3 12 1 12 14 1 9 8 5 2 8 2 5 2 1 14 4 2 14 14 1	7 2 7 4 5 6 7 7 4 7 6

Note

The picture in the statements illustrates the first example.

Problem C. Duathlon

Time limit: 1 second
Memory limit: 1024 megabytes

The Byteburg's street network consists of n intersections linked by m two-way street segments. Recently, the Byteburg was chosen to host the upcoming duathlon championship. This competition consists of two legs: a running leg, followed by a cycling leg.

The route for the competition should be constructed in the following way. First, three distinct intersections s, c, and f should be chosen for start, change and finish stations. Then the route for the competition should be built. The route should start in s, go through c and end in f. For safety reasons, the route should visit each intersection at most once.

Before planning the route, the mayor wants to calculate the number of ways to choose intersections s, c, and f in such a way that it is possible to build the route for them. Help him to calculate this number.

Input

The first line contains integers n and m: number of intersections, and number of roads. Next m lines contain descriptions of roads ($1 \le n \le 10^5$, $1 \le m \le 2 \cdot 10^5$). Each road is described with pair of integers v_i, u_i, the indices of intersections connected by the road ($1 \le v_i, u_i \le n$, $v_i \ne u_i$). For each pair of intersections there is at most one road connecting them.

Output

Output the number of ways to choose intersections s, c, and f for start, change and finish stations, in such a way that it is possible to build the route for competition.

Scoring

Subtask 1 (points: 5)

$n \le 10$, $m \le 100$

Subtask 2 (points: 11)

$n \le 50$, $m \le 100$

Subtask 3 (points: 8)

$n \le 100\,000$, there are at most two roads that ends in each intersection.

Subtask 4 (points: 10)

$n \le 1\,000$, there are no cycles in the street network. The cycle is the sequence of k ($k \ge 3$) distinct intersections $v_1, v_2, \dots v_k$, such that there is a road connecting v_i with v_{i+1} for all i from 1 to $k-1$, and there is a road connecting v_k and v_1.

Subtask 5 (points: 13)

$n \le 100\,000$, there are no cycles in the street network.

Subtask 6 (points: 15)

$n \le 1\,000$, for each intersection there is at most one cycle that contains it.

Subtask 7 (points: 20)

$n \le 100\,000$, for each intersection there is at most one cycle that contains it.

Subtask 8 (points: 8)

$n \leq 1\,000$, $m \leq 2\,000$

Subtask 9 (points: 10)

$n \leq 100\,000$, $m \leq 200\,000$

Examples

input	output
4 3 1 2 2 3 3 4	8
4 4 1 2 2 3 3 4 4 2	14

Note

In the first example there are 8 ways to choose the triple (s, c, f): $(1, 2, 3)$, $(1, 2, 4)$, $(1, 3, 4)$, $(2, 3, 4)$, $(3, 2, 1)$, $(4, 2, 1)$, $(4, 3, 1)$, $(4, 3, 2)$.

In the second example there are 14 ways to choose the triple (s, c, f): $(1, 2, 3)$, $(1, 2, 4)$, $(1, 3, 4)$, $(1, 4, 3)$, $(2, 3, 4)$, $(2, 4, 3)$, $(3, 2, 1)$, $(3, 2, 4)$, $(3, 4, 1)$, $(3, 4, 2)$, $(4, 2, 1)$, $(4, 2, 3)$, $(4, 3, 1)$, $(4, 3, 2)$.