5980 GPS Reverse Positioning

It's a misty day in Manhattan and you have lost your way. Driving your car, you are trying to figure out what your current location is. The foggy weather does not allow you to see any sign or information outside the car, and like other nerds, you hate asking for directions. But fortunately enough, you already have two other sources of information: a map of Manhattan, and a GPS/navigational module installed on your car. The bad news is that the module is malfunctioning due to high humidity and it does not accept any user inputs. Therefore, you can only use the information available on its current screen, which is a list of locations you had already entered on the system (called points of interest) each with the shortest distance (through the roads) from your current position to that location.

Given the map data and the information seen on the screen, your job is to find all locations on the map which are possibly your current position.

The map is modeled with N map nodes, and R two-way roads. Each node is a point on the map specified by its coordinates. Each road is a line segment on the map specified by the two nodes it connects, which are called its heads. As you might already know, all roads in Manhattan are either vertical or horizontal. A node might be the head of one or more roads. A car can go from a road r to a node n and vice versa (i.e. from n to r), if and only if the node n is a head of the road r. By definition, it is not possible to drive a car from a road directly to another road without using any intermediate nodes, even if the roads have a nonempty intersection, as the city has many non-planar structures. Similarly, you cannot drive from one node directly to another node without using any intermediate roads even if they have the same coordinate on the map. Though, it is guaranteed that you can drive from any node of the map to any other node using some intermediate roads and nodes.

The points of interest seen on the screen are also given as some nodes of the map. Though, your current position is not necessarily on a node of the map; in this case, you are on a road. The shortest-path distances given on the screen are calculated according to the rules specified above.

Input

The input contains several test cases. Each test case starts with a line containing three space-separated integers N ($1 \leq N \leq 10000$), the number of nodes, R ($1 \leq R \leq 10000$), the number of roads, and M ($1 \leq M \leq 100$), the number of the points of interest. The nodes are numbered $1, 2, \ldots, N$. The next N lines specify the coordinates of the nodes. The i-th line of this part contains two space-separated integers x_i and y_i ($-10^5 \leq x_i, y_i \leq 10^5$), the coordinates of node i. Then, R lines follow, each with two space-separated integers u_i and v_i, indicating that the two nodes u_i and v_i are connected by a road. It is guaranteed that the road is either vertical ($x_{u_i} = x_{v_i}$) or horizontal ($y_{u_i} = y_{v_i}$). The test case ends with M lines specifying the information available on the GPS system screen. Each of these lines has two space-separated integers w_i and d_i ($0 \leq d_i \leq 10^9$), where w_i is the number of a node as a point of interest, and d_i is the distance from your current position to node w_i.

The input ends with a line of the form '`0 0 0`' (omit the quotes) which should not be processed as a test case.

Output

For each test case, you should first write a line containing a single integer k, the number of distinct points on the map that might be your current position. Then, on each of the next k lines, you should write two space-separated integers X_i and Y_i as the coordinates of one of the possible points. The coordinates must be in ascending order based on their X_i's, and then by their Y_i's (when their X_i's are equal).

Sample Input

```
7 6 2
0 1
1 0
1 1
1 3
3 1
3 3
4 0
1 3
3 5
5 6
4 3
3 2
2 7
7 5
5 3
3 2 1
0 0
2 0
3 0
1 2
1 3
1 1
3 2 1
0 0
0 2
0 0
1 2
2 3
1 3
3 2 2
0 0
0 2
0 0
1 2
2 3
1 3
3 3
0 0
```

Sample Output

```
2
0 1
1 2
1
1 0
1
0 1
```

```
0
```