Many of you probably know Peter Longfoot, who was a student in University of Suburbia in 2005. He does not like crowded streets so in World Finals 2005 in Shanghai,contestants wrote a program for him that helped him find the path from his home to his university that required least crossing of streets. But in Suburbia streets were drawn as horizontal or vertical lines. Now Peter has become a student of University of Chingchuk. The streets of Chingchuk be can drawn as straight line but they are not necessarily horizontal or vertical as shown in picture below. Also not all streets are equally crowded. If there is a place near a street that can cause a lot of crowd (such as super markets or railway stations) then the road segments adjacent to it will become more crowded and so peter may want to avoid them also if possible.

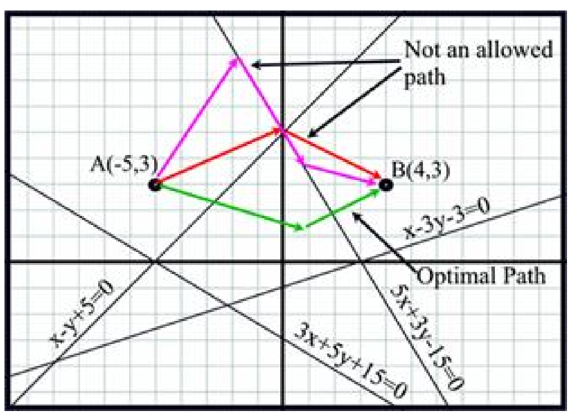

Figure 1: The green arrow shows one optimal path to go from A to B. The red arrow and magenta arrow show two possible invalid paths. The red path is invalid because walking through the crossing of two or more streets is not allowed and the magenta path is invalid because walking along the street is invalid.

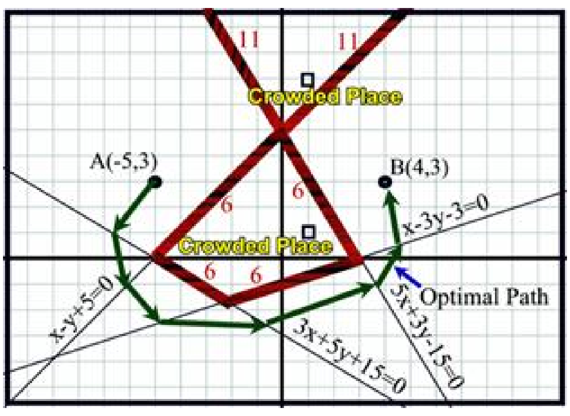

Figure 2: The square marked places are crowded places. The crowded place at (1,1) has crowding index 5 and the crowded place at (1,7) has crowding index 10. So now the new minimum cost path from A to B is shown in dark green. Red color denotes the streetss whose costs have been increased due to crowded places.

Figure 1 has four streets. If Peter's house is at A and his University is at B then he requires crossing minimum two streets. By default the cost of crossing a street is 1 so his total cost of reaching his university is 2. Peter never walks through a street crossing and also he never walks along the street. In Figure 2 there are two crowded places, one at C(1, 1) and the other at D(1,7). The crowding index of C is 5 and the crowding index of D is 10. So now the road segments adjacent to C has crossing cost (5+1)=6 and the road segments adjacent to D has crossing cost (1+10)=11. A road segment is adjacent to a crowded place if that road segment can be reached from the crowded place without crossing any other street. Given the road map of Chingchuk, the coordinate of Peter's home and university and the coordinate and crowding index of the crowded places your job is to find the minimum cost that Peter needs to reach his university.

Input

The input file contains at most 100 sets of inputs. Most cases are not extreme (N is much less than 40). The description of each set is given below:

First line of each set contains three integers N ($2 \leq N \leq 35$), C ($0 \leq C \leq 1000$) and Q ($0 \leq Q \leq 10$). The integer N indicates how many streets are there, C indicates the number of crowded places and Q indicates the total number of queries. Each of the next N lines contains three integers a_i, b_i, c_i ($1 \leq i \leq N$ and $0 \leq |1000a_i|, |1000b_i|, |c_i| \leq 1000000$ and both a and b will never be zero) which indicates that the equation of the i-th street is $a_i x + b_i y + c_i = 0$. Each of the next C lines contains three integers x_c, y_c, C_c. This means that there is a crowded place at location (x_c, y_c) and it increases the crossing costs of adjacent road segments by C_c, ($1 \leq C_c \leq 20$). Each of the next Q lines contains four integers x_h, y_h, x_u, y_u, which implies that for that specific query the coordinate of Peter's home is (x_h, y_h) and the coordinate of Peter's university is (x_u, y_u). You can assume that Peter's home, Peter's University and the crowded places are never situated on a street and also no two streets are parallel two each other. You can also assume that absolute values of all coordinates are not greater than 1000.

Input is terminated by a set where the value of $N = C = Q = 0$.

Output

For each set of input produce $Q + 1$ lines of output. The first line contains the serial of output and each of the next Q lines contains a single integer. This integer indicates the cost of going from (x_h, y_h) to (x_u, y_u).

Look at the output for sample input for formatting details.

Sample Input

```
4 0 1
1 -1 5
3 5 15
5 3 -15
1 -3 -3
-5 3 4 3
4 2 2
1 -1 5
3 5 15
5 3 -15
1 -3 -3
1 1 5
1 7 10
-5 3 4 3
1 9 4 3
4 3 4
1 -1 5
3 5 15
5 3 -15
1 -3 -3
1 1 5
1 7 10
1 8 18
-5 3 4 3
1 9 4 3
1 2 1 11
1 11 1 12
0 0 0
```

Sample Output

```
Case 1:
2
Case 2:
6
11
Case 3:
6
29
35
0
```