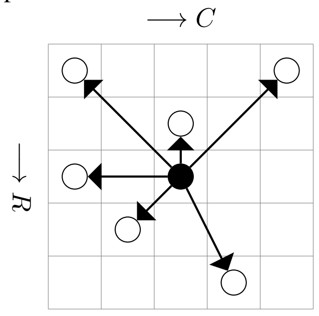
Problem JUMP3D: Jump3D

Jump3D is a 3D board game consisting of a stack of boards with L levels. Each level is a board with $R \times C$ squares in a grid with R rows and C columns. Every player starts from the same square on one of the boards. The goal is to reach a specific target square as quickly as possible. However, the assigned target square is different for each player. In each step, one of the following moves is possible:



Furthermore, it is possible to jump up *one* level or jump down *two* levels (increasing, resp. decreasing the level number) in one step. Jumping up and down does neither change row nor column. Finally, B squares are blocked, so it's not possible to move or jump on these. It is also not allowed to move/jump outside of a board.
Obviously, a random board configuration is unfair, as it may prefer some players. A configuration is considered fair if every player can reach its target square and the minimal number of steps from start to target is equal for every player. It is considered better when the minimal distance from start to targets is longer. What's the start square in the best fair configuration given the (already selected) target squares of all players?

Input

The first line of the input contains the integers L, R, and C. These describe the number of levels, rows, and columns ($0 < L * R * C \leq 50\,000$). The next line contains the integer P, the number of players ($0 < P \leq 10$). Then follow P lines, each containing three integers $l\ r\ c$, specifying the level, row and column of a player's target square ($0 \leq l < L$; $0 \leq b < B$; $0 \leq r < R$). The next line contains the integer B, the number of blocked squares ($0 \leq B \leq L*R*C-P$). Then follow B lines, each containing three integers $l\ r\ c$, specifying the level, row and column of a blocked square. You may safely assume that no square is blocked twice and no target square is blocked.

Output

If there is no fair board configuration, print "`UNFAIR`". Otherwise print two lines: the first specifying the start square as three integers (the level, row, and column) in the best board configuration. If there is more than one, print any. On the second line, print two integers: the minimal distance D from start to targets squares for this best board configuration and the number of possible start squares with distance $= D$.

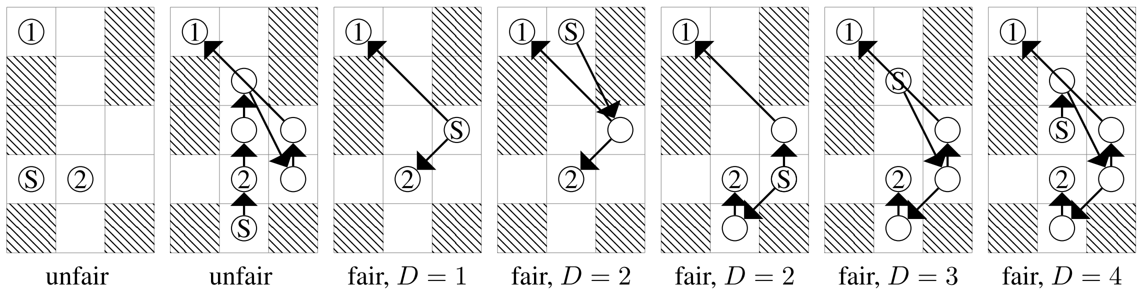


Figure 1: Illustration of first sample input: shows the shortest move sequences from each possible start square (S) to the target squares (1 resp. 2).

Sample Input 1

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

Sample Output 1

```
0 2 1
4 1
```

Sample Input 2

```
1 5 3
2
0 0 0
0 3 1
8
0 1 0
0 2 0
0 4 0
0 0 2
0 1 2
0 4 2
0 2 2
0 3 2
```

Sample Output 2

```
UNFAIR
```

Sample Input 3

```
30 30 30
2
0 29 29
23 0 0
2
1 29 29
0 27 28
```

Sample Output 3

```
0 12 0
29 2
```