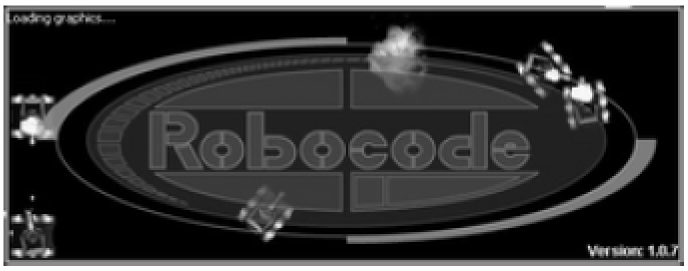
Problem H: Robocode



Robocode is an educational game designed to help learn Java. The players write programs that control tanks fighting with each other on a battlefield. The idea of this game may seem simple, but it takes a lot of effort to write a winning tank's program.

Today we are not going to write an intelligent tank, but to design a simplified robocode ***game engine***.

Assuming the whole battlefield is 120*120 (pixel). Each tank can ONLY move in vertical and horizontal direction on the fixed path (There are paths every 10 pixels in the battlefield in both vertical and horizontal direction. In all there are 13 vertical paths and 13 horizontal paths available for tanks, as shown in Figure 1).

The shape and size of the tank are negligible and one tank has (x, y) $(x, y \in \{0, 10, 20, ..., 120\})$ representing its coordinate position and α ($\alpha \in \{0, 90, 180, 270\}$) representing its facing direction (α = 0, 90, 180 or 270 means facing right, up, left or down respectively). They have a constant speed of 10 pixels/second when they move and they can't move out of the boundary (they will stop moving, staying in the direction that they are currently facing, when touching any boundary of the battlefield).

The tank can shoot in the direction it's facing whether it's moving or still. The shot moves at the constant speed 20 pixel/second and the size of the shot is also negligible. It will explode when it meets a tank on the path. It's possible for more than one shot to explode in the same place if they all reach a tank at the exact same time.

Two tanks being met on the path will be both destroyed and will be removed from the battlefield at once also. The tank being hit by the explosion will be destroyed and will be removed from the battlefield at once. A shot exploding or flying out of the boundary will also be removed.

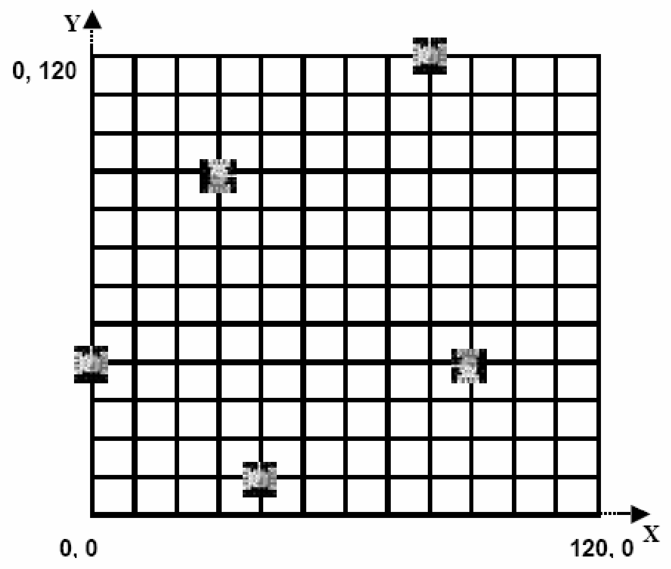


Figure 1

When the game begins, all the tanks are stopped at different crosses of the vertical and horizontal paths. Given the initial information of all the tanks and several commands, your job is to find the winner -- the last living tank when all the commands are executed (or omitted) and no shot exists in the battlefield (meaning that no tank may die in the future).

Input

There are several test cases. The battlefield and paths are all the same for all test cases as shown in Figure 1. Each test case starts with integers *N* (1 <= *N* <= 10) and *M* (1 <= *M* <= 1000), separated by a blank. *N* represents the number of the tanks playing in the battlefield, and *M* represents the number of commands to control the tanks' moving. The following *N* lines give the initial information (at time 0) of each tank, in the format:

Name x y α

The *Name* of a tank is consisted of no more than 10 letters. *x*, *y*, *α* are integers and *x*, *y* Є {0, 10, 20, ..., 120}, *α* Є {0, 90, 180, 270}. Each field is separated by a blank.

The following *M* lines give commands in such format:

Time Name Content

Each field is separated by a blank. All the commands are giving in the ascending order of *Time* (0 <= *Time* <= 30), which is a positive integer meaning the timestamp when the commands are sent. *Name* points out which tank will receive the command. The *Content* has different types as follows:

MOVE	When receiving the command, the tank starts to move in its facing direction. If the tank is already moving, the command takes no effect.
STOP	When receiving the command, the tank stops moving. If the tank is already stopped, the command takes no effect.
TURN *angle*	When receiving the command, the tank changes the facing direction *α* to be ((*α* + *angle* + 360) mod 360), no matter whether it is moving or not. TURN command doesn't affect the moving state of the tank.
SHOOT	When receiving the command, the tank will shoot one shot in the direction it's facing.

Tanks take the corresponding action as soon as they receive the commands. E.g., if the tank at (0, 0), *α* = 90, receives the command MOVE at time 1, it will start to move at once and will reach (0, 1) at time 2. Notice that a tank could receive multiple commands in one second and take the action one by one. E.g., if the tank at (0, 0), *α* = 90, receives a command sequence of "TURN 90", "SHOOT", "TURN -90", it will turn to the direction *α* = 180, shoot a shot and then turn back. If the tank receives a command sequence of "MOVE; STOP", it will keep still in the original position.
Some more notes you need to pay attention:
♦ If a tank is hit by an explosion, it will take no action to all the commands received at that moment. Of course, all the commands sent to the already destroyed tank should also be omitted.
♦ Although the commands are sent at discrete seconds, the movement and explosions of tanks and shots happen in the continuous time domain.
E.g. lets test case includes the following program:

```
2 2
A 0 0 90
B 0 10 0
10 A SHOOT
10 B MOVE
```

As a result, tank B escapes from A's shoot because B leaves (0,10) just after time 10 while the shoot doesn't reach (0,10) yet.

A test case with *N* = *M* = 0 ends the input, and should not be processed.

Output

For each test case, output the winner's name in one line. The winner is defined as the last living tank. If there is no tank or more than one tank living at the end, output "NO WINNER!" in one line.

Sample Input

```
2 2
A 0 0 90
B 0 120 180
1 A MOVE
2 A SHOOT
2 2
A 0 0 90
B 0 120 270
1 A SHOOT
2 B SHOOT
2 6
A 0 0 90
B 0 120 0
1 A MOVE
2 A SHOOT
6 B MOVE
30 B STOP
30 B TURN 180
30 B SHOOT
0 0
```

Output for the Sample Input

```
A
NO WINNER!
B
```