A robot has been sent to explore a remote planet. To specify the path the robot should take, a program is sent each day. The program consists of a sequence of the following commands:

- FORWARD: move forward by one unit.
- TURN LEFT: turn left by 90 degrees. The robot remains at the same location.
- TURN RIGHT: turn right by 90 degrees. The robot remains at the same location.

The robot also has sensor units which allows it to obtain a map of its surrounding area. The map is represented as a grid of M rows and N columns. Each grid point is represented by a coordinate (r, c) where $r = 0$ is the north edge of the map, $r = M - 1$ is the south edge, $c = 0$ is the west edge, and $c = N - 1$ is the east edge. Some grid points contain hazards (e.g. craters) and the program must avoid these points or risk losing the robot.

Naturally, if the initial location and direction of the robot and its destination position are known, we wish to send the shortest program (one consisting of the fewest commands) to move the robot to its destination (we do not care which direction it faces at the destination). You are more interested in knowing the number of different shortest programs that can move the robot to its destination, because we may need to send different sequences as interplanetary communication is not necessarily reliable. However, the number of shortest programs can be very large, so you are satisfied to compute the number as a remainder under some modulus, knowing that something you learned in classes called the Chinese remainder theorem can be used to compute the final answer.

Input

The input consists of a number of cases. The first line of each case gives three integers M, N, and the modulus m ($0 < M, N \leq 1000$, $0 < m \leq 1000000000$). The next M lines contain N characters each and specify the map. A '`.`' indicates that the robot can move into that grid point, and a '`*`' indicates a hazard. The final line gives four integers r_1, c_1, r_2, c_2 followed by a character d. The coordinates (r_1, c_1) specify the initial position of the robot, and (r_2, c_2) specify the destination. The character d is one of '`N`', '`S`', '`W`', '`E`' indicating the initial direction of the robot. It is assumed that the initial position and the destination are not hazards. The input is terminated when $m = 0$.

Output

For each case, print its case number, the modulus, as well as the remainder of the number of different programs when divided by the modulus m. The output of each case should be on a single line, in the format demonstrated below. If there is no program that can move the robot to its destination, output '`-1`' for the number of different programs.

Sample Input

```
3 3 100
***
.*.
***
1 0 1 2 E
4 4 100
****
*.*.
*.*.
*...
1 1 1 3 N
4 8 100
********
...**...
*......*
********
1 0 1 7 E
0 0 0
```

Sample Output

```
Case 1: 100 -1
Case 2: 100 2
Case 3: 100 4
```