Ants on a plane!

In this étude you are going to be simulating (some of) the behaviour of creatures related to Langton's ant. Imagine an (in principle) infinite plane, with a lonely ant initially standing at $(0, 0)$. The ant will take a certain sequence of steps of unit length in one of the four compass directions determined by various rules (i.e., its "DNA"). These rules specify the direction of the next step based on the previous step, and the state of the ant's current position – they may also specify a change in that state.

For instance, the traditional Langton's Ant lives on a plane with two possible states (black and white) and with the rules "right on white, left on black, flip the colour".

The basic problem you will be asked to solve here is, given an ant's DNA, determine its location after a certain number of steps (starting on a plane whose points are all of some given state).

Task

One of the purposes of this étude is to introduce a standard format we will be using for many tasks. In this format, input will come from `stdin` and will consist of a sequence of "scenarios" separated from one another by blank lines. There may also be some lines beginning with the hash character, #. These are comments and should be ignored entirely. The output for each scenario is to echo the input for that scenario (not including comments), followed by a line beginning with #, followed by a single space, and then the required "answer" for the scenario[1].

In this task an individual scenario consists of the DNA of an ant, followed by a positive integer (the number of steps you are to follow the ant for). For Langton's ant, the DNA is represented as follows:

```
w ESWN bbbb
b WNES wwww
```

Each line of DNA consists of three parts:

- A single character representing a state.
- A sequence of four compass directions representing the direction of the next step from a point in this state, after arriving with a North, East, South, and West step respectively (so if a Langton's ant arrives at a white point from the south, i.e., using a North step, its next step is East – a right turn).

[1] Can you see what I did here? In testing I can use the correct output file as input, and then just `diff` your output with it – no need to keep separate input and output files around!

- A sequence of four characters representing the state of the point after the ant leaves (again based on the incoming direction).

Initially, all points of the plane are presumed to be in the state coded by the first line of the DNA (e.g., `w` above), and the ant is facing North (i.e., its first step is to be determined as if it had arrived at (0,0) from the south).

It is *not* to be assumed that there will always be just two states, or that they are coded by letters. For instance this is a perfectly valid DNA representation:

```
. EEEE xxxx
y SEWN x.!y
x WEEE !!!!
! NWES .yx!
```

You *may* assume that the DNA is always complete (i.e., you will not be given possible states in the third part that are not represented in the list).

The output for a scenario is simply a # character, followed by a space, then the x-coordinate of the ant's final position, another space, and the y-coordinate of the ant's final position.

(Pair 2)