Problem 3—Frogger

Konami came out with the arcade game "Frogger" in 1981. One unusual feature was that each Frogger arcade unit had 2 CPUs. The gameplay involved guiding your frog across a busy highway and a dangerous river to make it home. This problem relates to the highway portion of the game.

Each input case depicts a highway with one or more lanes of traffic. Vehicles in the uppermost lane (row) move towards the right side of the screen, vehicles in the next lane move towards the left, and so on, with each row alternating direction. All vehicles wrap around to the other side upon reaching the edge of the screen. Both Frogger and the vehicles move one square per time step, but alternate in doing so (Frogger moves first, then all vehicles move, then frogger moves again, etc.). Unlike traffic, frogger cannot wrap around the sides of the screen (nor exit from the bottom). At each time step, Frogger can move horizontally, vertically or stay still, but cannot move diagonally. You are to determine whether or not it is possible for Frogger to escape the highway by jumping off the top of the screen, and if so, the least number of time steps this requires.

INPUT SPECIFICATION.

You will be given a set of input cases. Each case consists of two positive integers, separated by one space, giving the width and height (i.e. length and number of lanes) making up the highway. This is followed by a visual depiction of the highway, where vehicles are represented with * and spaces indicate the lack of any vehicle.
Frogger's starting position is indicated with an F.
"0 0**<EOLN>**" will follow the last case.

OUTPUT SPECIFICATION.

For each case, your are to output the minimum number of moves it takes to get Frogger to jump off of the top of the screen, followed by **<EOLN>**. If Frogger is doomed to collide with traffic, output splat“**<EOLN>**”. If Frogger is neither doomed nor can make it home, output trapped “**<EOLN>**”.

SAMPLE INPUT

```
6·2<EOLN>
*·**·*<EOLN>
··F···<EOLN>
6·2<EOLN>
*·**·*<EOLN>
*·F···<EOLN>
6·2<EOLN>
*··*·*<EOLN>
··F···<EOLN>
6·2<EOLN>
****··<EOLN>
··F···<EOLN>
6·3<EOLN>
****··<EOLN>
****··<EOLN>
··F···<EOLN>
0·0<EOLN>
<EOF>
```

SAMPLE OUTPUT

```
trapped<EOLN>
splat<EOLN>
2<EOLN>
4<EOLN>
5<EOLN>
<EOF>
```