D. Broken Monitor

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Innocentius has a problem — his computer monitor has broken. Now some of the pixels are "dead", that is, they are always black. As consequence, Innocentius can't play the usual computer games. He is recently playing the following game with his younger brother Polycarpus.

Innocentius is touch-typing a program that paints a white square one-pixel wide frame on the black screen. As the monitor is broken, some pixels that should be white remain black. Polycarpus should look at what the program displayed on the screen and guess the position and size of the frame Innocentius has painted. Polycarpus doesn't like the game but Innocentius persuaded brother to play as "the game is good for the imagination and attention".

Help Polycarpus, automatize his part in the gaming process. Write the code that finds such possible *square* frame that:

- the frame's width is 1 pixel,
- the frame doesn't go beyond the borders of the screen,
- all white pixels of the monitor are located on the frame,
- of all frames that satisfy the previous three conditions, the required frame must have the smallest size.

Formally, a square frame is represented by such pixels of the solid square, that are on the square's border, that is, are not fully surrounded by the other pixels of the square. For example, if the frame's size is $d = 3$, then it consists of 8 pixels, if its size is $d = 2$, then it contains 4 pixels and if $d = 1$, then the frame is reduced to a single pixel.

Input

The first line contains the resolution of the monitor as a pair of integers n, m $(1 \le n, m \le 2000)$. The next n lines contain exactly m characters each — the state of the monitor pixels at the moment of the game. Character "`.`" (period, ASCII code 46) corresponds to the black pixel, and character "`w`" (lowercase English letter `w`) corresponds to the white pixel. It is guaranteed that at least one pixel of the monitor is white.

Output

Print the monitor screen. Represent the sought frame by characters "`+`" (the "plus" character). The pixels that has become white during the game mustn't be changed. Print them as "`w`". If there are multiple possible ways to position the frame of the minimum size, print any of them.

If the required frame doesn't exist, then print a single line containing number `-1`.

Examples

input

```
4 8
..w..w..
........
........
..w..w..
```

output

```
..w++w..
..+..+..
..+..+..
..w++w..
```

input

```
5 6
......
```

```
.w....
......
..w...
......
```

output

```
......
+w+...
+.+...
++w...
......
```

input

```
2 4
....
.w..
```

output

```
....
.w..
```

input

```
2 6
w..w.w
...w..
```

output

```
-1
```

Note

In the first sample the required size of the optimal frame equals 4. In the second sample the size of the optimal frame equals 3. In the third sample, the size of the optimal frame is 1. In the fourth sample, the required frame doesn't exist.