B Bee Problem

You are a busy little bee, and you have a problem. After collecting nectar all day long, you are returning to the beehive with a large supply of honey. You would really like to take a nap now, but sadly, you have to store all the honey in your beehive first. Opening up a cell in the hive to funnel honey into takes a lot of time, so you want to avoid doing this as much as possible.

Some of the cells in the beehive are already filled with old, hardened honey. The other cells are still empty. If you pour honey into an empty cell, it will automatically start flowing into adjacent empty cells. From these cells, the honey will again flow to other neighbouring empty cells. This saves you from having to funnel honey into them directly. You decide to use this to your advantage, by pouring into cells with lots of (indirect) adjacent open cells.

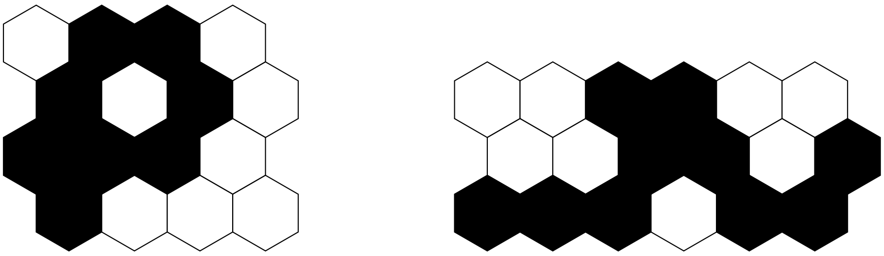

Figure 1: The beehives from the first two samples. The black hexagons already contain hardened honey. The white cells are still empty.

You have some units of honey, and know the layout of your beehive. By cleverly choosing which cells to funnel honey into, what is the minimal amount of work you have to do?

Input

- The input starts with three integers, $0 \leq h \leq 10^6$, the amount of honey you have, and $1 \leq n, m \leq 10^3$, the dimensions of the grid.
- Then follow n lines, one for each row of the grid. Each row has m symbols, either `.`, representing an empty cell, or `#`, representing a filled cell. These symbols are separated by spaces. Furthermore, every second row starts with a space, as these are slightly offset to the right.

The grid always has enough open cells to store all your honey.

Output

Output a single integer, the number of cells you have to funnel honey into directly to store all your honey in the hive.

Sample Input 1

```
8 4 4
. # # .
 # . # .
# # # .
 # . . .
```

Sample Output 1

```
3
```

Sample Input 2

```
6 3 6
. . # # . .
 . . # # . #
# # # . # #
```

Sample Output 2

```
2
```