Jumping Rooks

HackerRank

Nina has an $n \times n$ chessboard and k jumping rooks. Every cell of the chessboard is either *blocked* or *free*, and Nina can only put a *single* rook in any *free* cell.

Two jumping rooks beat each other if they are either in the same row or in the same column *and* all cells between them are free (note that it's possible that there are some other rooks between them). More formally, if the first rook is in cell (x, y_1) and the second rook is in cell (x, y_2) (where $y_1 \leq y_2$), then these two rooks beat each other if and only if $(x, y_1), (x, y_1 + 1), \ldots, (x, y_2)$ are free. If the rooks are in cells (x_1, y) and (x_2, y), then cells $(x_1, y), (x_1 + 1, y), \ldots, (x_2, y)$ must all be free.

Given the configuration of the chessboard and some k, help Nina place k jumping rooks in the chessboard's free cells such that the number of pairs of rooks that beat each other is minimal. Then print a single integer denoting the number of rooks that beat each other.

Input Format

The first line contains two space-separated integers describing the respective values of n (the size of the chessboard) and k (the number of rooks to place).
Each line i of the n subsequent lines contains a string of n characters describing each row in the chessboard. The j^{th} character of the i^{th} line is `#` if cell (i, j) is blocked or `.` if the cell is free.

Constraints

- $1 \leq n \leq 50$
- It is guaranteed that k is less than the number of free cells in the chessboard.

Output Format

Print a single integer denoting the minimum possible number of pairs of rooks that beat each other.

Sample Input 0

```
3 4
...
...
...
```

Sample Output 0

```
2
```

Explanation 0
For this input, one possible arrangement is:

```
o.o
.o.
..o
```

where each `o` is a jumping rook.

Sample Input 1

```
5 10
..#..
..#..
#####
..#..
..#..
```

Sample Output 1

```
4
```

Explanation 1

For this input, one possible arrangement is:

```
.o#o.
oo#oo
#####
.o#o.
o.#.o
```

where each `o` is a jumping rook.