HCI '16 - Jumping

Time limit: 0.35s (Java: 1.5s) **Memory limit:** 64M (Java: 256M)

Pepe the frog needs to jump across a river while only stepping on some stones! He needs to jump from column 1 to column N (left to right in the input). Rock i means the rock on row i. However, jumping from row i to j takes $(i-j)^2$ time.

Input

The first line of the input contains two integers, N and C, the number of columns and the number of positions of rocks.

Each subsequent C lines of the input contains a string of N `0`s and `1`s. A `1` on line i (starting from 2nd line) on column j (both 0-indexed) indicates the rock j is present in column i, a `0` indicates that rock j is not present.

Output

Output the minimum amount of time taken to jump to the end.

You are guaranteed there is at least one possible way to jump to the end.

Constraints

Subtask 1 [10%]

$1 \leq N, C \leq 7$

Subtask 2 [5%]

$1 \leq N, C \leq 100$

Subtask 3 [20%]

$1 \leq N, C \leq 500$

Subtask 4 [20%]

$1 \leq N, C \leq 750$

Subtask 5 [45%]

$1 \leq N, C \leq 2\,000$

Subtask 6 [0%]

Sample test cases.

Sample Input 1

```
5 3
11100
00011
11111
```

Sample Output 1

```
0
```

Explanation for Sample Output 1

You don't need to jump, just use rock 3 throughout.

Sample Input 2

```
5 5
10000
01000
00100
00010
00001
```

Sample Output 2

```
4
```

Explanation for Sample Output 2

Start from rock 0, change to rock 1, change to rock 2, etc. Each of them takes 1 time because they are adjacent.