D. Snake

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Let us remind you the rules of a very popular game called "Snake" (or sometimes "Boa", "Python" or "Worm").

The game field is represented by an $n \times m$ rectangular table. Some squares of the field are considered impassable (walls), all other squares of the fields are passable.

You control a snake, the snake consists of segments. Each segment takes up exactly one passable square of the field, but any passable square contains at most one segment. All segments are indexed by integers from 1 to k, where k is the snake's length. The 1-th segment is the head and the k-th segment is the tail. For any i $(1 \le i < k)$, segments with indexes i and $i + 1$ are located in the adjacent squares of the field, that is, these squares share a common side.

One of the passable field squares contains an apple. The snake's aim is to reach the apple and eat it (that is, to position its head in the square with the apple).

The snake moves throughout the game. During one move the snake can move its head to an adjacent field square. All other segments follow the head. That is, each segment number i $(1 < i \le k)$ moves to the square that has just had segment number $i - 1$. Consider that all segments including the head move simultaneously (see the second test sample). If the snake's head moves to an unpassable square or to the square, occupied by its other segment, the snake dies. That's why we will consider such moves unvalid.

Your task is to determine the minimum number of valid moves that the snake needs to reach the apple.

Input

The first line contains two space-separated integers n and m $(1 \le n, m \le 15)$ — the number of rows and columns of the game field.

Next n lines describe the game field. Each of these lines contains m characters. Character "#" represents a wall, "." is a passable square, "@" is an apple. The snake's first segment is represented by character "1", the second one segment — by character "2" and so on.

The game field description doesn't contain any characters besides "#', ".", "@" and digits (except 0). It is guaranteed that the described field is correct. It is guaranteed that the described field contains exactly one apple and exactly one snake, the snake's length is at least 3 and at most 9.

Output

Print a single integer to the output — the minimum number of moves needed to reach the apple. If the snake can't reach the apple, print -1.

Examples

input

```
4 5
##...
..1#@
432#.
...#.
```

output

```
4
```

input

```
4 4
#78#
.612
.543
..@.
```

output

```
6
```

input

```
3 2
3@
2#
1#
```

output

```
-1
```