Captain Qs Treasure

English Vietnamese

You got an old map, which turned out to be drawn by the infamous pirate "Captain Q". It shows the locations of a lot of treasure chests buried in an island.

The map is divided into square sections, each of which has a digit on it or has no digit. The digit represents the number of chests in its 9 neighboring sections (the section itself and its 8 neighbors). You may assume that there is at most one chest in each section.

Although you have the map, you can't determine the sections where the chests are buried. Even the total number of chests buried in the island is unknown. However, it is possible to calculate the minimum number of chests buried in the island. Your mission in this problem is to write a program that calculates it.

Input

The input is a sequence of datasets. Each dataset is formatted as follows. h w map The first line of a dataset consists of two positive integers h and w. h is the height of the map and w is the width of the map. You may assume 1 <= h <= 15 and 1 <= w <= 15.

The following h lines give the map. Each line consists of w characters and corresponds to a horizontal strip of the map. Each of the characters in the line represents the state of a section as follows.

- '.': The section is not a part of the island (water). No chest is here.
- '*': The section is a part of the island, and the number of chests in its 9 neighbors is not known.
- '0' .. '9': The section is a part of the island, and the digit represents the number of chests in its 9 neighbors.

You may assume that the map is not self-contradicting, i.e., there is at least one arrangement of chests. You may also assume the number of sections with digits is at least one and at most 15. A line containing two zeros indicates the end of the input.

Output

For each dataset, output a line that contains the minimum number of chests. The output should not contain any other character.

Example

Input:
5 6
*2.2**
..*...
..2...
..*...
*2.2**

6 5
.*2*.
..*..
..*..
..2..
..*..
.*2*.
5 6
.1111.
**...*
33....
**...0
.*2**.
6 9
....1....
...1.1...
....1....
.1..*..1.
1.1***1.1
.1..*..1.
9 9
****
*4*4*4*4*
****
*4*4*4*4*
****
*4*4*4*4*
****
*4*4*4***
****
0 0

Output:
6
5
5
6
23