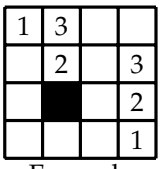
Logical reasoning and programming, task I

(October 24, 2017)

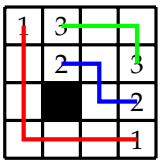
Problem

Your task is to produce a solver for a puzzle called Numberlink — the goal is to find paths that connect the matching characters in a grid. Assume you have a grid, in our case always $n \times n$, and some positions are already filled with characters. There are pairs of matching characters that you are supposed to connect. However, standard rules are slightly modified. Some squares are forbidden[1], say contain walls, they are already occupied and cannot be used, see the black square.



Example

You have to connect each matching pair of characters with a single continuous path. A path starts in one of the characters and ends in the other one. Moreover, path are not allowed to branch off or cross over each other. In our example, the following is the only possible solution.



Solution

For simplicity, you can assume that an input satisfies the following. First, there is at most one solution; it is possible that no solution exists. Second, if a solution exists, then all the empty squares have to be used. On the other hand, do not assume that an input contains walls (zeros).

Program

You should upload an archive that contains an executable script `numberlink` that expects an input string on `stdin` and produces a solution to `stdout`.

It is expected that you use Python (use `python2` or `python3`), but MATLAB 9.2 (use `matlab`) should also work. You can use

- MiniSat, command `minisat`,
- PicoSAT, command `picosat`,

[1] This is our modification of standard Numberlink.

- PycoSAT in Python, `import pycosat`,

as solvers. You are allowed to use other solvers included in your archive.

Every input has a maximal time for which you can solve it, however, the given time should be more than enough for solving the problem using a decent SAT solver with a non-optimized encoding. In principle, you can use other techniques to solve the task, but be aware of these timeouts.

Non-standard settings can be discussed individually.

Input

An input is a string of length $n \times n$. In our example it is

`...1.0.2.2.313..`

where `.` is an empty square, `0` is a wall, and `1...9` and `A...Z` are characters that have to be connected. A square (x, y) is described by a character at the position $(n \cdot y) + x$ in the string, we start counting from zero.

Output

The output of your solver is again a string with empty squares filled with characters that correspond to paths going through them.

1	3	3	3
1	2	2	3
1	■	2	2
1	1	1	1

Filled grid

Hence you are supposed to produce

`1111102212231333`

If no solution is possible, then just produce string

`0`