## *Logical reasoning and programming*, task I

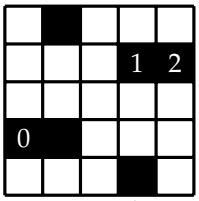
**(October 16, 2023)**

### Problem

Your task is to produce a solver for a puzzle called Light Up (or Akari) using a SAT solver.

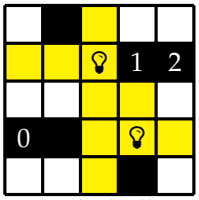
You have a grid, in our case always $n \times n$, with white and black cells. The goal is to place light bulbs in white cells so that all white cells are lit. A light bulb illuminates all white cells visible horizontally and vertically, but all black cells (walls) block light. Moreover, no light bulb is allowed to illuminate another light bulb (but more light bulbs may illuminate a cell). Black cells may contain a number (0, 1, 2, 3, or 4) that specifies the exact number of light bulbs occurring in the adjacent cells (diagonal neighbors do not count).

For example, if we have the following assignment,



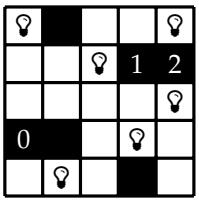
Example

we may place two light bulbs as follows.



Light bulb

They illuminate the yellow cells; one of the cells is illuminated by both light bulbs. By adding more light bulbs, we obtain the only possible solution:



Solution

For simplicity, you can assume that there is always at most one solution; no solution may exist.

## Program

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

It is expected that you use Python 3.8.3 (use `python3`), but MATLAB 9.13 (use `matlab`) should also work. You may use

- PySAT in Python (including cardinality constraints), `import pysat`,
- PycoSAT in Python, `import pycosat`,
- MiniSat, command `minisat`,
- PicoSAT, command `picosat`,

as SAT solvers. You are allowed to use another SAT solver included in your archive.

Every input has a maximum execution time assigned. However, the given time should be enough for solving the given problem using any of the previous solvers with a decent (non-optimized) encoding.

Non-standard settings should be discussed individually.

### Input

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

```
WWWBW0BWWWWWWWWWW12WBWWW
```

where `W` is a white cell, `B` is a black cell without a number, and `0`...`4` are black cells containing the corresponding number. A cell $(x, y)$ 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 the string you receive on the input where your placement of light bulbs is indicated by replacing the corresponding `W`s by `L`s.

Hence the correct output for our example is

```
WLWBW0BWLWWWWWLWWL12LBWWL
```

If no solution is possible, then just produce string

```
0
```

## Points

Your score is assigned automatically by BRUTE; you also see some test cases, including the one presented here. However, only slightly incorrect code may get very little points. For that reason, your code will be manually reviewed after the deadline, and additional points may be awarded.

*Please, do submit even incomplete solutions!*