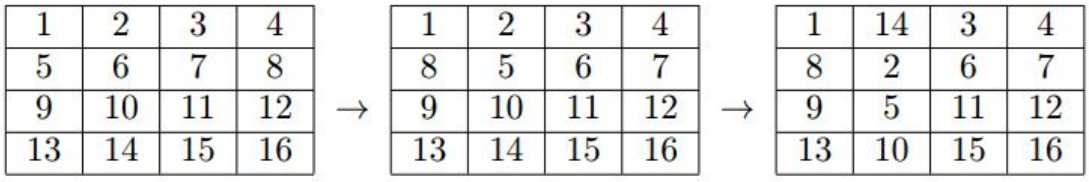
Consider a variant of the 15-puzzle, but with the following important changes. First, instead of 15 tiles, there are 16, so that there are no empty spaces on the board. Second, instead of moving a single tile into an open space, a move in this puzzle consists of either (a) sliding an entire row of tiles left or right, with the left- or right-most tile “wrapping around” to the other side of the board, or (b) sliding an entire column of the puzzle up or down, with the top- or bottom-most tile “wrapping around.” For example, here is a sequence of two moves on such a puzzle:



The goal of the puzzle is to find a short sequence of moves that restores the canonical configuration (on the left above) given an initial board configuration.

solution to this problem found efficiently using A* search.
Program should run on the command line like: python solver16.py [input-board-filename] where input-board-filename is a text file containing a board configuration in a format like:
5 7 8 1
10 2 4 3
6 9 11 12
15 13 14 16

output format: [move-1] [move-2] ... [move-n] where each move is encoded as a letter L, R, U, or D for left, right, up, or down, respectively, and a row or column number (indexed beginning at 1). For instance, the two moves in the picture above would be represented as: R2 D2