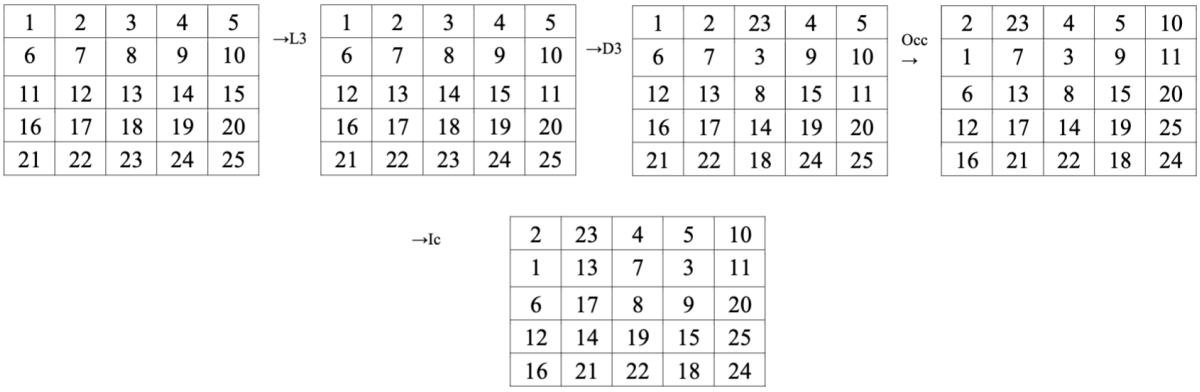
The 2023 Puzzle

Consider the 2023 puzzle, which is a lot like the 8-puzzle we talked about in class, but: (1) it is 5 x 5 with 25 tiles, so there are no empty spots on the board; (2) 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 one space, with the left- or right-most tile 'wrapping around' to the other side of the board, (b) sliding an entire column of tiles up or down one space, with the top- or bottom-most tile 'wrapping around' to the other side of the board, (c) rotating the outer 'ring' of tiles either clockwise or counterclockwise, or (d) rotating the inner ring either clockwise or counterclockwise.

For example, here is a sequence of three 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. We've provided skeleton code to get you started. You can run the skeleton code on the command line:

python3 solver2023.py [input-board-filename]

where input-board-filename is a text file containing a board configuration (we have provided an example). You'll need to complete the function called solve(), which should return a list of valid moves. The moves should be encoded as strings in the following way:

- For sliding rows, R (right) or L (left), followed by the row number indicating the row to move left or right. The row numbers range from 1-5.
- For sliding columns, U (up) or D (down), followed by the column number indicating the column to move up or down. The column numbers range from 1-5.
- For rotations, I (inner) or O (outer), followed by whether the rotation is clockwise (c) or counterclockwise (cc).

For example, the above diagram performs the moves L3 (slide row 3 left), D3 (slide column 3 down), Occ (outer counterclockwise), and Ic (inner clockwise).