Key Algorithms

Draw Line Logic:

The key functionality of the app is to let user draw lines by connecting dots in the user drawing area. It is achieved by the following functions:

drawLine(cell1, cell2): the function draws the line between two dots, which are passed through the function arguments.

LoadPatternAndConnect(patternDots): the function renders the example pattern by taking the example pattern in the form of dots in sequence and draw lines between each adjacent dot.

RedrawPatternWhenScroll(patternDots): the function relocate user drawn lines when there is scroll event or resize event by redraw them in the current screen size and scroll position configuration.

ReallignCells(): the function relocates both example pattern and user drawn lines by first remove lines in both example pattern and user drawing area and redraw them in the current screen size and scroll position configuration.

Checker Logic:

After the test taker clicks the submit button, the app will call the function “checkCorrectness” in checker.js. This function will take three inputs and return a boolean value(true/false). The first input is the original pattern, which is an array of integers ranging from 1 to 16. The second is a string indicating the task requirement of the current page, for example, “copy”, “lateral”, “vertical”, etc. The last parameter is the user input, whose data type is the same as the original pattern. The system will convert what the test taker has drawn into an array of integers ranging from 1 to 16.

Inside the function “checkCorrectness”, we will first convert the data type of the original pattern and user input into an array of two-element-tuple (row, column) coordinates, named original path and user path. Then the system will compute the target pattern according to the task requirements. After that, compare the target with the user path, if they are all the same, return true to tell that the test taker has successfully finished the subtask, otherwise it will give false.

The method of computation:

Copy:

The target is just a copy of the original path.

Lateral flip:

After lateral flip, the target is generated by using a loop to transform each coordinate from (Ri, Ci) to (Ri, 5 - Ci) according to the original order. Row remains unchanged. The points in the first column will be flipped to the fourth column, and the points in the second column will be flipped to the third column. We can observe that the sum of the indices of the old and new columns is always 5.

Vertical flip:

After vertical flip, the target is generated by using a loop to transform each coordinate from (Ri, Ci) to (5 - Ri, Ci) according to the original order. Column remains unchanged. The points in the first row will be flipped to the fourth row, and the points in the second row will be flipped to the third row. We can observe that the sum of the indices of the old and new rows is always 5.

Rotate 180 degrees:

After a 180-degree-rotation, the target is generated by using a loop to transform each coordinate from (Ri, Ci) to (5 - Ri, 5 - Ci) according to the original order. A 180-degree-rotation is equivalent to a lateral flip followed by a vertical flip.

Flip over y = x:

After a flip over the line y = x, the target is generated by using a loop to transform each coordinate from (Ri, Ci) to (5 - Ci, 5 - Ri) according to the original order. We observe this rule by seeking the regularity, (1,2) map to (3,4), (2,2) map to (3,3), (2,4) map to (1,3), (1,1) map to (4,4)......