**COMP1206 MathDoku Instructions**

**Guide:** This document will help us run and use your application during marking. Please complete the sections below. You may want to include screenshots if this helps explain the functionality. For most sections, 1-2 sentences are probably sufficient.

If you did not implement a particular part, please write “not implemented” in the relevant section.

These instructions are not assessed directly, but they will help ensure that we do not miss any important features of your application.

|  |
| --- |
| **Installing and Running the Application (Part 1)**  *Copy and paste the contents of your README.txt file below.* |
|  |
| **Starting a Game (Optional – Part 1)**  *If any additional steps are needed to start a game, briefly describe them here.* |
| A game must either be generated or loaded, as when the application starts there is no game to be played. |
| **Cell Completion (Part 3)**  *Describe how to enter and clear cell values by keyboard and by mouse.* |
| By Keyboard:  Type a value between 1 and N into the cell you select.  By Mouse:  Select a cell by clicking on it, then click on the button with the number between 1 and N on the right-hand side of the board to enter that number in the selected cell. |
| **Can your application handle - and ÷ cages with more than two cells? (Part 4)** |
| Yes |
| **Mistake Detection (Part 4)**  *Describe how to enable mistake detection in your application.* |
| By selecting the check box “Show Mistakes” at the bottom of the application. This can be turned off by unchecking the check box. |
| **Win Detection / Animation (Parts 4 & 8)**  *Describe how the application notifies the player when the game is won (including any animations you have implemented for Part 8).* |
| The user is notified when the game is won by the winning animation starting immediately when the final number is placed in the last cell. After the animation is complete the user is notified with an alert congratulating them for completing the puzzle.  Animation:   * A sequence of 8 different colours snaking down the grid row by row, starting in the top left and zigzagging down, side to side. * Once complete this sequence then continues but instead zigzags up and down from one side to another. * Once this is complete the sequence then spirals towards the centre of the grid. * Once the sequence completes spiralling into the centre of the grid, the colours make their way outwards from the centre in an explosion effect. |
| **Clearing (Part 5)**  *Describe how to clear the board.* |
| Clicking the clear button at the bottom of the application. The user must the select the “OK” button to confirm they wish to clear the board on the alert.  Note: if the board is already empty an alert will appear informing the user there is nothing to clear. |
| **Undo/Redo (Part 5)**  *Describe how to undo / redo actions.* |
| Undo: clicking the “Undo” button on the left of the “Clear” button at the bottom of the application. If there are no possible moves to undo, then this button is disabled and cannot be clicked.  Redo: clicking the “Redo” button on the right of the “Clear” button at the bottom of the application. If there are no possible moves to redo, then this button is disabled and cannot be clicked.  Note: These buttons both work with the clearing functionality i.e. a clear can be undone and then redone. |
| **Loading Files (Part 6)**  *Describe how to load puzzles both from file and through text input. Also mention any limitations in what puzzles you can load (if any), e.g., up to a certain size if smaller than 8x8.* |
| From File:  Clicking the “Load Game - File” button at the top of the application. This opens a popup file explorer which allows the user to select a text file to load.  From Text:  Clicking the “Load Game - Text” button at the top of the application. This opens a popup with a TextArea allowing the user to enter their puzzle. The user then clicks “Load” to load the puzzle onto the board.  Limitations (optional):  I have limited the maximum grid size to be 10.  Puzzle must be in the correct format or message will appear informing the user the file/text is not in the correct format. |
| **Font Sizes (Part 7)**  *Describe how to change font sizes* |
| Clicking the “Font Size” button at the top right of the application. This opens a popup with 5 different options for the font size (tiny, small, medium (the default), large & huge). The user simply clicks on their choice and the font is changed for the board. The current font size is highlighted so the user knows which of the 5 options is selected already. |
| **Solver (Part 9)**  *Describe how to solve a puzzle, how to get a hint and any limitations there might be (e.g., up to what size you can solve reliably and within <1 min). Also mention where we can find your code for solving the puzzle (which files and lines)?* |
| Solve puzzle:  Not implemented  Get hint:  Limitations (optional):  Files / lines for solver: |
| **Random Game Generator (Part 10)**  *Describe how to generate a random game, including what options the player can select. Also specify where we can find your code for generating the puzzle (which files and lines)? Where in the code do you ensure there is only one solution (which file and lines)?* |
| Generate puzzle (including options):  Clicking the “Generate Random Game” at the top of the application. A popup will appear with a TextField that the user can enter the size of the board (limited to 10 at most) for the generated game. The user can also select a difficulty for the game that will be generated. If no difficulty is selected it defaults to “Medium”. The user the clicks the “Generate” button which then will generate the puzzle.  There is no guarantee that the puzzle generated has a unique solution.  Files / lines for generator:  RandomGenerator.java – all lines  File / lines to ensure there is only one solution:  Not implemented |
| **Additional Information (Optional)**  Any other information that may be useful for us to know. |
|  |