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.*
To run the program, simply compile "javac Game.java" before running "java Game". This should open the game window.
Starting a Game (Optional – Part 1) *If any additional steps are needed to start a game, briefly describe them here.*
Load a game through pressing: (a) Load game from file button – then select the desired .txt file (b) Load game from text button – input text in correct format and press ok Files defining game grids ranging from 2x2 to 8x8 are accepted.
Cell Completion (Part 3) *Describe how to enter and clear cell values by keyboard and by mouse.*
By Keyboard: Select cell using mouse (or cycle through with “tab” keyboard button) and enter number from keyboard – backspace works to delete an inputted number. By Mouse: Select cell using mouse, then press one of the buttons on the right-hand side, labelled 1 to 8 respectively. The buttons do not steal focus from the selected cell so you can choose a different number for the cell without having to reselect the 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.*
Pressing the “show mistakes” button highlights any rows, columns or cages that are incorrect. In order to update which cells should be highlighted, please press the “show

mistakes" button to turn off mistake detection before re-enabling it (it does not dynamically update as you change cell values).
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).*
There are no animations to alert the winner – the player is notified that they have won through a message alert as soon as the correct grid is entered onto the screen.
Clearing (Part 5) *Describe how to clear the board.*
Pressing the "clear" button will result in a confirmation alert pop-up, from here you can either cancel or continue with clearing the board. This can be undone through pressing the "undo" button until all cells are the same as before. The clear button is disabled when the board is already clear.
Undo/Redo (Part 5) *Describe how to undo / redo actions.*
The undo and redo buttons are disabled when there are no actions to redo or undo. Pressing the "undo" button will undo the last change made to the grid, you can continue undo-ing until the board is back to its original clear state. Pressing "redo" will redo any changes made by the previous undo button, you can continue redo-ing until all previous undo commands are re-done.
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.*
Load a game through pressing: (a) Load game from file button – then select the desired .txt file (only txt files are able to be selected) (b) Load game from text button – input text in correct format and press ok Files defining game grids ranging from 2x2 to 8x8 are accepted. The program will check that the text file is valid in terms of cells in cages being adjacent to one another and whether cells are part of only one cage each – a warning alert will display if the file is not valid.
Font Sizes (Part 7)

Describe how to change font sizes
//
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: 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): Files / lines for generator: File / lines to ensure there is only one solution:
Additional Information (Optional) Any other information that may be useful for us to know.