# Scope

Currently, the program includes a problem generator, file reader/writer, and a simple monster Sudoku solver that uses backtracking search.

The problem generator creates a random monster Sudoku puzzle out of parameters initial amount of solved cells, size of the puzzle, column size per block, and row size per block.

The file reader/writer reads in input files to either generate Sudoku puzzles or if there is already a Sudoku puzzle in the input file, the reader create a usable Sudoku object for the solver. The reader can also write to an output file a randomly generated Sudoku file.

The solver can solve a random monster Sudoku puzzle using backtracking search. Currently, it does not use any heuristics or constraint propagation. The solver creates a log file after solving, containing useful information on the process.

# Progress

* Created and implemented a fully functional Problem Generator that generates a random monster Sudoku puzzle using input parameters.
* Created and implemented a fully functional Sudoku file reader and writer that reads in and create Sudoku puzzles from files and writes puzzles to files.
* Created and implemented a functional Sudoku solver using only backtracking search.

# Problems & Questions

* Currently, it is unknown whether it can be run in the openlab environment.

# Results

The solver was tested with various randomly generated Sudoku puzzles, ranging from to . Something I have noticed was that the solver using only backtracking search would often time out with , even with variable . As expected, as , the puzzle was less and less likely to be solvable, something the solver figured out rather quickly.

# Notes

* I have implemented a problem generator in C++ and a file reader/writer without using the provided Java shell.
* I have also implemented my own solver using backtracking search without using the provided Java shell.