

Status of Completion

Initial Problem Statement:

In short, the program shall solve and generate a standard 9x9 Sudoku Grid using some basic logical and guessing (whenever required) techniques. Thus the problem is split into two linked but different parts –

A. In case of solving a user-given incomplete Grid:

1. If an invalid Sudoku grid, i.e. one which is unsolvable due to inconsistencies in the given digits, an appropriate error message explaining the user the problem will be displayed.
2. The program will feature an “activity box” which if activated by the user will slow down the algorithm, to show each step of solving to help a novice understand and learn the basic solving logic for a Sudoku.
3. A pencil marks scheme (if possible to be quickly and elegantly coded within the remaining time) may be implemented. Please see the attached gui_scheme.pdf file for further details.
4. Saving and loading a puzzle from a file will also be implemented.
5. This functionality can be used for checking a Grid sourced from elsewhere or to get some hints into its solving.

B. In case of generating a random, new and incomplete Grid for the user to solve:

1. A difficulty level analysis shall be implemented based on the number of guesses required (higher weightage) and the number of times the logic is run (less weightage) to solve the Sudoku.
2. To make it more interesting and give it a slight twist, instead of a numerical Sudoku, various pictures – such as logos of popular brands, images of fruits, etc. – will replace numbers if the user wants so (such an option will be given).
3. A timer will also be present for the user to track his speed and proficiency in solving.
4. Again, hints will be provided for the user to get some further insight into the puzzle solving.

Final Status – Achieved parts are highlighted green, unused parts are highlighted grey and the shelved ideas are highlighted red or blue respectively.

A.2. The activity box code is ready and working but commented out (It can be easily made to appear on terminal). On the GUI, due to the relatively ugly font used by RenderText, we decided that it would be best to keep the code unused as the consistency of the interface would be completely thrown off balance.

A.3. The pencil marks scheme was construed to be too exhausting to code using EzWindows and hence the idea was shelved.

B.1. Due to the limitations of the algorithm (use of backtracking) it is very difficult to gauge a sensible difficulty level of any Sudoku. Hence, we decided to make merely a basic one based on the number of blanks – which, frankly, is NOT a good primary criterion. Hence, though the code is done and working, it is not used.