Fun with Solvers
– An exercise in programming –

Wilhelm Simus

September 17, 2019

As an exercise in programming I want to implement solvers for some commonly known riddles starting with a Sudoku solver. The result shall be a tool generating these riddles for me to play as leisure activity.

This document shall contain the outline of the project with tasks organized in the order I intended to be following.

Every few tasks are grouped in a milestone. I use Git's Issue and Milestone system to track them on GitHub:

- ☐ **1. All Set Up**
  - ☑ Decide on programming language
  - ☐ Create a manual
  - ☐ Input - How to process it
- ☐ **2. Necromancy - Create a Skeleton**
  - ☐ Design the UI
  - ☐ Implement the Main Menu
  - ☐ Implement the Input Windows for each kind of Riddle
  - ☐ Implement Game Windows for each Riddle
- ☐ **3. It's something**
  - ☐ Implement Sudoku Solver
  - ☐ Implement Kakuro Solver
- ☐ **4. Here's a riddle, have some fun**
  - ☐ Implement Sudoku Generator
  - ☐ Implement Kakuro Generator
  - ☐ Implement Shikaku Generator
- ☐ **5. I ran all the numbers**
  - ☐ Implement Shikaku Solver
  - ☐ Implement SumSum Solver
- ☐ **6. Smooth Generator**
  - ☐ Implement SumSum Generator
  - ☐ Implement Picross Generator
- ☐ **7. I want to play a game! C:**
  - ☐ Implement Picross Solver

Documentation

1. All Set Up

Decide on programming language

The options are

- Python
- C++
- Java

Results:
After a colleague hinted me towards GTK I decided to use Python for the backend and the GTK library (PyGObject) for implementation of the GUI.

Create a manual

I want to create a manual explaining all five types of riddles. It shall be written in LaTeX and it should contain a reasonable amount of images helping with the explanation. For every riddle a minimal example shall be demonstrated to maximize understandability.

Results:

Input - How to process it

How do I put a riddle into the solver program?
Ideally the program should have an interface where one can chose the type of quiz: You click the quiz you want, select it's size (only relevant for quizzes with variable size) and fill out the clues. Then you click on a button and the solver generates a solution.

The idea of this part is to think about and research (efficient) ways in which I can represent the riddles within the program (on the GUI aswell as in the backend). This way my tasks will be more clear in the implementation-phase.

Results:

2. Necromancy - Create a skeleton

Design the UI

In this part I do not implement anything yet: I create a document in which I imagine how I want the program to look and feel (which button does what, what options do I have, etc...). I document this process to have an outline of what is to be implemented in the next three steps. The only thing I already decided on is three windows:

- a Main Menu from which I can chose which mode of the program I want to use
- a window for the Solver mode of the program
- a window for the Game mode / Generator mode of the program

The documentation should be detailed enough to be implemented.

Results:

Implement the Main Menu

The Main Menu with all it's functions shall be implemented as documented before.

Results:

Implement the Input Windows for each kind of riddle

The Solver Mode Windows with all it's functions shall be implemented as documented before.

Results:

Implement Game Windows for each riddle

The Game Mode Windows with all it's functions shall be implemented as documented before.

Results:

3. It's something

Implement Sudoku Solver

A solver for the Sudoku riddle shall be implemented so that games that are put into the Solver Mode or generated within the Game Mode can be solved by the program.

Results:

Implement Kakuro Solver

A solver for the Kakuro riddle shall be implemented so that games that are put into the Solver Mode or generated within the Game Mode can be solved by the program.

Results:

4. Here's a riddle, have some fun

Implement Sudoku Generator

A generator for the Sudoku riddle shall be implemented so that games can be played in Game Mode.

Results:

Implement Kakuro Generator

A generator for the Kakuro riddle shall be implemented so that games can be played in Game Mode.

Results:

Implement Shikaku Generator

A generator for the Shikaku riddle shall be implemented so that games can be played in Game Mode.

Results:

5. I ran all the numbers

Implement Shikaku Solver

A solver for the Shikaku riddle shall be implemented so that games that are put into the Solver Mode or generated within the Game Mode can be solved by the program.

Results:

Implement SumSum Solver

A solver for the SumSum riddle shall be implemented so that games that are put into the Solver Mode or generated within the Game Mode can be solved by the program.

Results:

6. Smooth Generator

Implement SumSum Generator

A generator for the SumSum riddle shall be implemented so that games can be played in Game Mode.

Results:

Implement Picross Generator

A generator for the Picross riddle shall be implemented so that games can be played in Game Mode.

Results:

7. I want to play a Game! C:

Implement Picross Solver

A solver for the Picross riddle shall be implemented so that games that are put into the Solver Mode or generated within the Game Mode can be solved by the program.

Results: