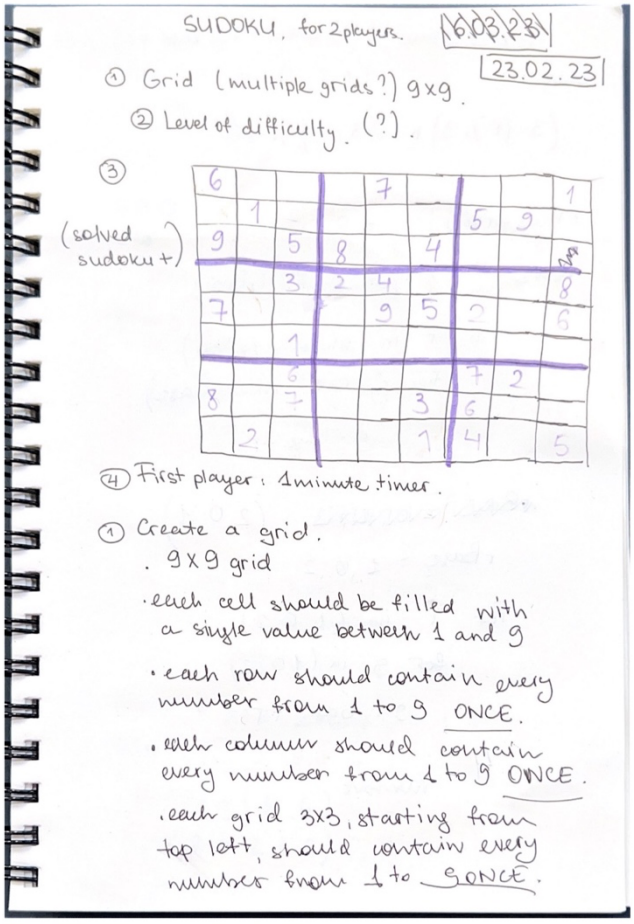
Sudoku for 2 players

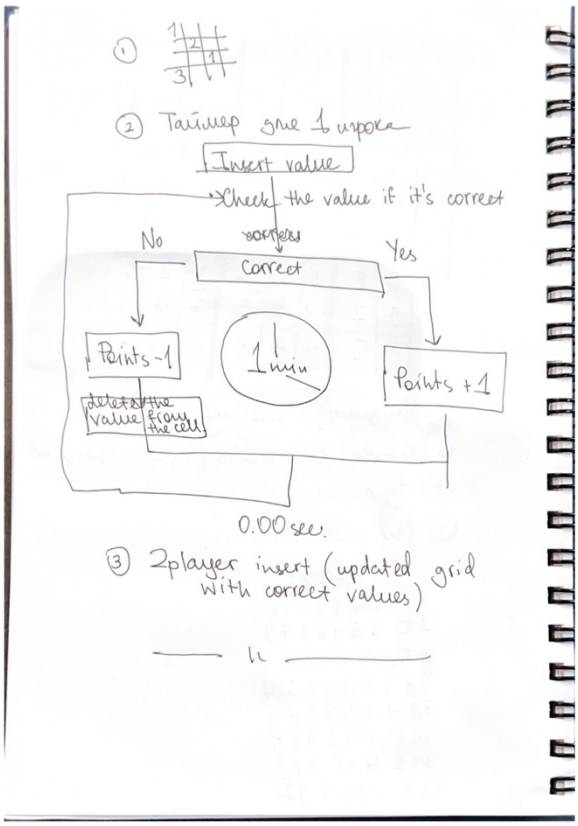
Rules

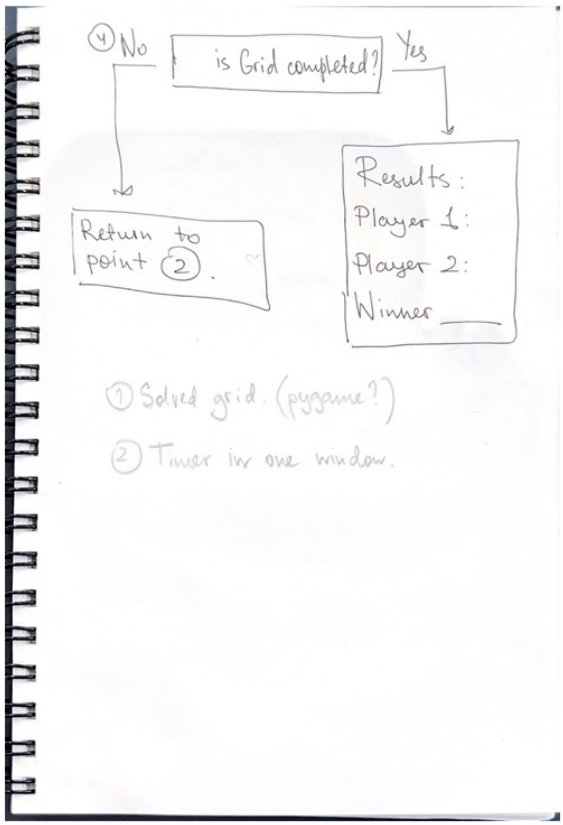
The idea behind the game was to create a sudoku grid that 2 players can solve at the same time. The rules are pretty simple: before the game starts, players can choose the level of the initial sudoku grid. When the game starts, each player has a 1-minute limit to fill in the grid. If the value is correct, then it stays on the grid and the score of the players goes high by 1. If the value is incorrect, the grid does not change, and the player's score goes down. The game ends when the grid is completed. The player that has the highest score, wins.



1 stage of game development

During the first stage of game development, I was trying to figure out how to structure my code. I knew that I would visualize the grid by using pygame, but I didn't dive into the interface development. More important was to understand the idea behind sudoku and how I can create an initial grid that will have only 1 solution. I read some articles about the math behind sudoku and how I can create initial grids that will be solvable. (LaBonte, Addison, "The Mathematics Behind Sudoku and How to Create Magic Squares" (2016). Honors College. 393.). I tried to write the code that will create an initial sudoku, but I stumbled upon problems that I could not solve with my capacities. When the solvable completed sudoku is created, what's hard is to delete the cells so that the initial grid would have 1 solution. It is also hard to create the function that will solve the grid as it needs backtracking and I could not manage to solve that problem. Luckily, I came across the python library





which is called **dokusan** that generates the initial grid with different levels of difficulties and solutions for them.

2 stage of game development

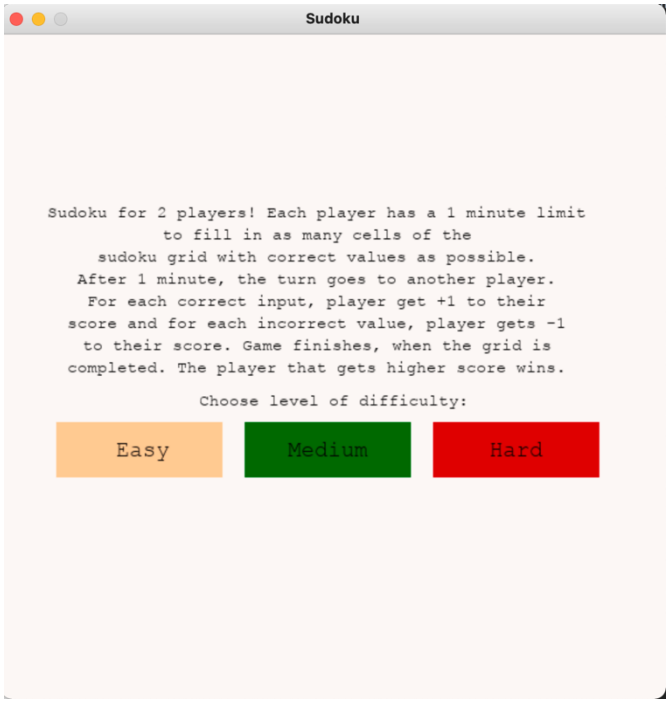
Once I figured out how I am going to create the sudoku girds, I started to think about how I am going to structure the logic of the game.

After creating the algorithm that represents the rules of the game, I started implementing it in code. To make my code readable, I created 2 files: functions and solver. In the functions file, I put all of the functions of the game (creating the window with gridlines, getting the initial grid and the completed one, extracting the position of the mouse on the grid when a player is selecting the cell, putting numbers in the cells with different colors, creating the timer, creating the score, highlighting the chosen cell). In the solver function, I put the main code that will run smoothly with help of the functions.

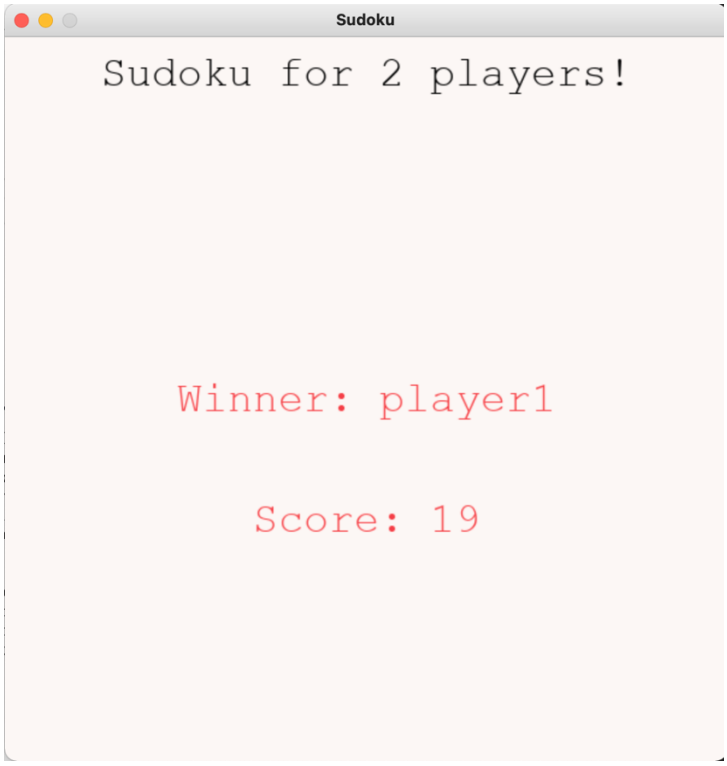
To debug code and find if it has any problems, I always printed the variables that I was working with to see the dynamic. Whenever I encountered a problem, it was always easy

to track where things have gone wrong by just printing all of the changes in the terminal. I also used some python language to test the code for errors and handle them (try-except).

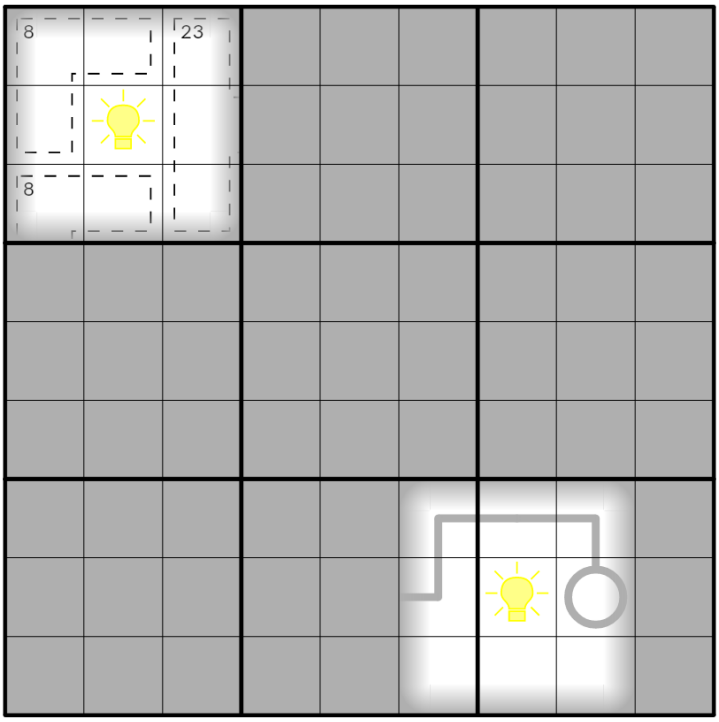
3 stage of game development



When the code has been finished, I started to clean it and make it more readable and adding comments. I also made some visual changes: I added the menu window that describes the rules of the game and made the game itself more flexible for players: at the beginning they can choose the difficulty level of the grid to challenge themselves.



Future implementations



In terms of future implementations of the game, I believe that it is interesting to dive into the math behind the sudoku and make the game multiplayer where the players can compete against the computer on who is going to solve the grid faster. It is also interesting to look at different versions of initial grids of sudoku (e.g. lumos maxima sudoku, where the grid is covered with fog and two light sources illuminate the darkness and clear the fog and the player should fill in the grid to get more light, see the picture below)

Packages that are used in the code

1. Dokusan, Dokusan.boards
2. Pygame
3. Copy

Git repository

I also added my coursework on GitHub which can be found at: https://github.com/alkiki/sudoku_for2

References

1. LaBonte, A. (no date) *The Mathematics Behind Sudoku and How to Create Magic Squares*. Available at: https://digitalcommons.library.umaine.edu/honors/393/.
2. *Lumos Maxima — Rätselportal — Logic Masters Deutschland* (no date). Available at: https://logic-masters.de/Raetselportal/Raetsel/zeigen.php?id=000BIU.