

Sudoku game

C++ OOP Version

Updates:

- OOP implementation
- Two classes implemented (one friend class)

```
generator.hpp > GeneratedGrid
6
7 class GeneratedGrid {
8     private:
9         int availArr[81][9]; //to generate a solvable 9*9 grid
10
11     protected:
12         int generatedGrid[9][9];
13
14     public:
15         GeneratedGrid();
16         //~GeneratedGrid();
17         int getNum (int row, int col);
18         void setNum (int row, int col, int num); //it should be guaranteed that the number set is valid (both location and value)
19
20         void printGrid();
21
22         //bool spaceAvailable (int row, int col);
23
24         //void initialAvailble();
25         //Given the blank Sudoku grid, the linear fill function will fill in every number into the block
26         //and ensures the filled numbers are legal so that the puzzle is solvable.
27
28         void linearFill();
29
30         bool outOfNumber(int indexAvail);
31         //Determine if there is still an available number in the list
32
33         void replenishSquareNum(int index);
34         //Remove the "count" number of digits from the generated solvable grid in a random manner.
35
36         void removeKDigits();
37
38         bool checkMoveLegal(int userDigit, int userRow, int userCol);
39         bool repeatInSubGrid (int userDigit, int userRow, int userCol);
40         bool repeatInRow (int userDigit, int userRow);
41         bool repeatInCol (int userDigit, int userCol);
42
43         friend class GameGrid;
44     };
45
46
47 #endif
```

```
game_mode.hpp > ...
1 #ifndef _GAME_MODE_H_
2 #define _GAME_MODE_H_
3
4 #include "generator.hpp"
5 #include "stdlib.h"
6 #include "string"
7 #include "fstream"
8
9 class GameGrid {
10     private:
11         int gameGrid[9][9];
12
13     public:
14         GameGrid(GeneratedGrid *gGrid);
15         //~GameGrid();
16
17         void playColour();
18         void resetColour();
19         void printGrid(GeneratedGrid *gGrid);
20
21         bool gameNotDone();
22         void playing(GeneratedGrid *gGrid);
23         bool notWantToContinue();
24
25         void gameRecord(GeneratedGrid *gGrid);
26         void clearRecord();
27 };
28
29
30 #endif
31
```

First Run

```
Please enter a number to choose the difficulty level (1 (easy), 2 (medium), 3 (hard): 1
  A  B  C  D  E  F  G  H  I
A  1  5  6 | 2  7  8 | 4  9  3 |
B  8  4  9 | 1  5  3 | 7  6  2 |
C  3  7  2 | 6  9  4 | 8  1  5 |
-----
D  4  9  7 | 5  6  2 | 1  3  8 |
E  6  3  1 | 8  4  7 | 2  5  9 |
F  2  8  5 | 9  3  1 | 6  7  4 |
-----
G  9  2  4 | 7  1  5 | 3  8  6 |
H  7  6  3 |   8  9 | 5  2  1 |
I  5  1  8 | 3  2  6 | 9  4  7 |
-----
If you want to restart this game, enter 10.
Exiting the game without finishing it may lead to the loss of data.
Enter the number you want to fill in: 4
Enter the location you want to fill 4 in: (row col): a 1

Invalid location! Please make sure the row and col are capital letters in the range of 'A' to 'I' (inclusive)

Enter the number you want to fill in: p
Invalid input!
Please enter a number from 1 to 9 (inclusive), or 10 to restart the game

Enter the number you want to fill in: 4
Enter the location you want to fill 4 in: (row col): A G

The move is illegal.
Please refer to the rules of sudoku to fill in a valid number

Enter the number you want to fill in: 4
Enter the location you want to fill 4 in: (row col): H D

  A  B  C  D  E  F  G  H  I
A  1  5  6 | 2  7  8 | 4  9  3 |
B  8  4  9 | 1  5  3 | 7  6  2 |
C  3  7  2 | 6  9  4 | 8  1  5 |
-----
D  4  9  7 | 5  6  2 | 1  3  8 |
E  6  3  1 | 8  4  7 | 2  5  9 |
F  2  8  5 | 9  3  1 | 6  7  4 |
-----
G  9  2  4 | 7  1  5 | 3  8  6 |
H  7  6  3 | 4  8  9 | 5  2  1 |
I  5  1  8 | 3  2  6 | 9  4  7 |
-----
You win!
```

- Different difficulty levels result in different number of digits removed
- (for simplicity of demo only one digit is removed)
- Problematic inputs will be caught. The program will tell the user what is wrong with the input
- User inputs have different color to be distinguished.
- Once the board is filled, the single game ends.

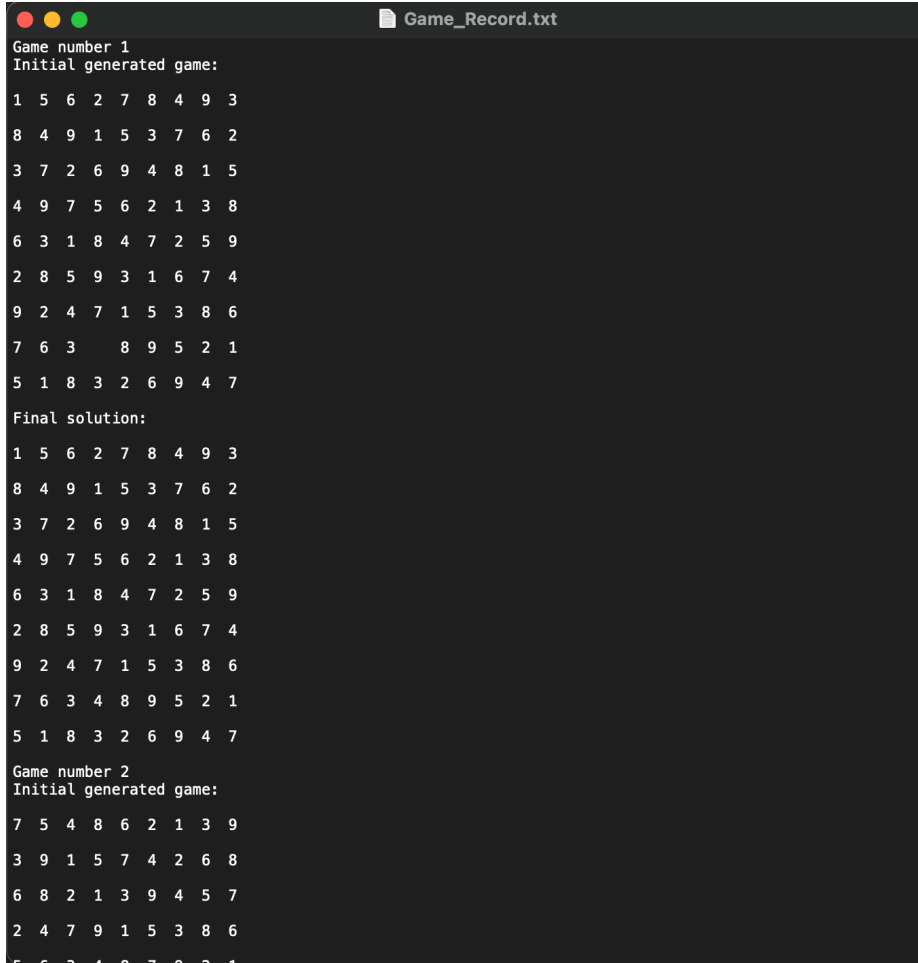
Second Run

```
Do you want to continue to the next game?
If you want to clear the past game data, enter 'c'. (y/n/c):
y
Please enter a number to choose the difficulty level (1 (easy), 2 (medium), 3 (hard): 1
  A  B  C  D  E  F  G  H  I
A  7  5  4 | 8  6  2 | 1  3  9 |
B  3  9  1 | 5  7  4 | 2  6  8 |
C  6  8  2 | 1  3  9 | 4  5  7 |
-----
D  2  4  7 | 9  1  5 | 3  8  6 |
E  5  6  3 | 4  8  7 | 9  2  1 |
F  9  1  8 | 6  2  3 | 7  4  5 |
-----
G  8  2   | 7  4  6 | 5  1  3 |
H  1  3  5 | 2  9  8 | 6  7  4 |
I  4  7  6 | 3  5  1 | 8  9  2 |
-----
If you want to restart this game, enter 10.
Exiting the game without finishing it may lead to the loss of data.
Enter the number you want to fill in: 9
Enter the location you want to fill 9 in: (row col): G C
  A  B  C  D  E  F  G  H  I
A  7  5  4 | 8  6  2 | 1  3  9 |
B  3  9  1 | 5  7  4 | 2  6  8 |
C  6  8  2 | 1  3  9 | 4  5  7 |
-----
D  2  4  7 | 9  1  5 | 3  8  6 |
E  5  6  3 | 4  8  7 | 9  2  1 |
F  9  1  8 | 6  2  3 | 7  4  5 |
-----
G  8  2  9 | 7  4  6 | 5  1  3 |
H  1  3  5 | 2  9  8 | 6  7  4 |
I  4  7  6 | 3  5  1 | 8  9  2 |
-----
You win!

Do you want to continue to the next game?
If you want to clear the past game data, enter 'c'. (y/n/c):
n
```

- After one game, the program asks if the user wants to continue.
- Three options are provided, respectively yes/no/clear.
- If y is entered, a new game is generated.
- If n is entered, the program stops.

Game Record File

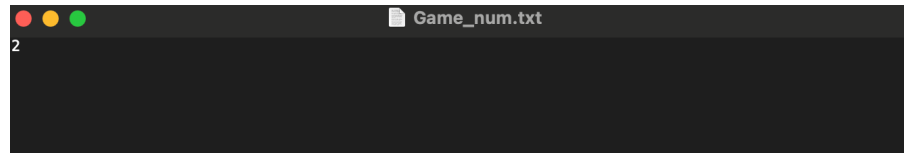


```
Game number 1
Initial generated game:
1 5 6 2 7 8 4 9 3
8 4 9 1 5 3 7 6 2
3 7 2 6 9 4 8 1 5
4 9 7 5 6 2 1 3 8
6 3 1 8 4 7 2 5 9
2 8 5 9 3 1 6 7 4
9 2 4 7 1 5 3 8 6
7 6 3 8 9 5 2 1
5 1 8 3 2 6 9 4 7

Final solution:
1 5 6 2 7 8 4 9 3
8 4 9 1 5 3 7 6 2
3 7 2 6 9 4 8 1 5
4 9 7 5 6 2 1 3 8
6 3 1 8 4 7 2 5 9
2 8 5 9 3 1 6 7 4
9 2 4 7 1 5 3 8 6
7 6 3 4 8 9 5 2 1
5 1 8 3 2 6 9 4 7

Game number 2
Initial generated game:
7 5 4 8 6 2 1 3 9
3 9 1 5 7 4 2 6 8
6 8 2 1 3 9 4 5 7
2 4 7 9 1 5 3 8 6
5 6 3 4 8 7 9 2 1
```

- At this point, the Game_Record.txt file saves the previous games played with game number recorded so that the user can keep track of the previous games.
- Even when the program restarts, the game number is still saved in another text file for the program to read and update.

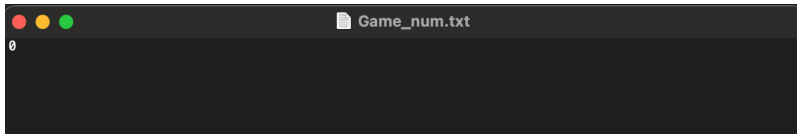


```
2
```

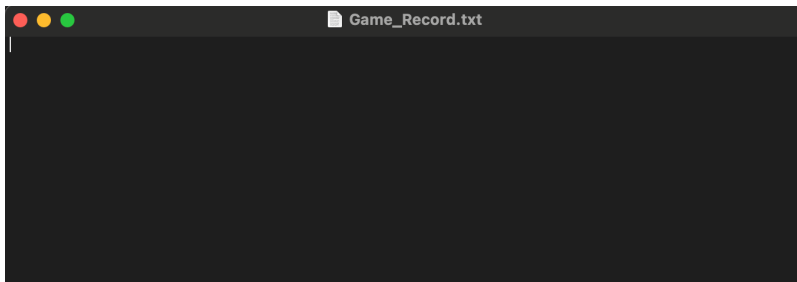
Clear record

- When the user enters 'c' to clear the record:
- Game_num resets to 0
- Game_Record resets to nothing

```
Do you want to continue to the next game?  
If you want to clear the past game data, enter 'c'. (y/n/c):  
c
```



A screenshot of a text editor window titled "Game_num.txt". The window has a dark background and a light-colored title bar with standard macOS window controls (red, yellow, green buttons). The main text area is dark, and the number "0" is visible in the top left corner.



A screenshot of a text editor window titled "Game_Record.txt". The window has a dark background and a light-colored title bar with standard macOS window controls (red, yellow, green buttons). The main text area is dark, and a single vertical cursor line is visible on the left side.