Project 1

Title

Sudoku Game

Course

CIS-5

Section

40404

Due Date

February 3, 2024

Author

Thu Pham

Introduction:

Welcome to the Sudoku Game! This program allows you to play the classic Sudoku puzzle. Sudoku is a number-placement puzzle that involves filling a 9x9 grid with digits from 1 to 9. The objective is to fill each row, each column, and each of the nine 3x3 sub grids (called "boxes") with all of the digits from 1 to 9.

Game Play and Rules:

To play the Sudoku game using this program, follow these steps:

1. Run the program.
2. Enter the initial values for the Sudoku puzzle. Use numbers from 1 to 9 to represent the known values and use 0 for empty cells that need to be filled.
3. The program will display the initial Sudoku puzzle with a visual representation of the 3x3 boxes.
4. The puzzle will be solved by the program. The empty cells will be filled with random numbers between 1 and 9.
5. The program will display the solved Sudoku puzzle.

V1:

1. **Include Libraries:**
    - The code includes standard C++ libraries for input/output (iostream), random number generation (cstdlib), and time functions (ctime).
2. **Set Random Seed:**
    - The srand function is used to set a random seed based on the current time. This ensures different random numbers each time the program runs.
3. **Declare Variables:**
    - Puzzle is declared to represent the 9x9 Sudoku grid.
4. **Initialize Variables:**
    - The program prompts the player to enter the Sudoku puzzle. The player is expected to input numbers for each cell of the 9x9 grid.

V2:

1. **Declare and Initialize Variables:**
    - After receiving input, the program displays the entered Sudoku puzzle.
2. **Display Entered Sudoku Puzzle:**
    - The code contains a loop to display the Sudoku puzzle that the player entered. It prints each row of the puzzle, separating the numbers with spaces.

V3:

1. **Solve Sudoku:**
    - The program enters a loop to solve the Sudoku puzzle. It iterates through each cell of the puzzle and, if the cell is empty (0), fills it with a random number between 1 and 9 using **rand()%9+1**.
2. **Display Solved Sudoku:**
    - After solving the Sudoku puzzle, the program displays the updated grid. It prints each row of the solved puzzle, separating the numbers with spaces.

V4:

The primary difference between V3 and V4 is the inclusion of a title, providing a clearer indication that the player is interacting with a Sudoku game. The core logic and functionality remain the same in both versions.

V5:

The Sudoku-solving approach in this code is based on filling empty cells with random numbers. More advanced solving algorithms, such as backtracking, could be implemented for a valid solution. This code provides a simple Sudoku game where the player can input an initial puzzle, and the program attempts to solve it by filling in the empty cells with random numbers.

Summary:

Project size: About 90+ lines

The number of variables: 2

The code includes basic input validation to check if the user-input values for the Sudoku puzzle are within the valid range (0 to 9). However, the program does not check for the validity of the initial Sudoku puzzle (whether it follows Sudoku rules). The program only runs once, displaying the initial Sudoku puzzle and its solved version.

Flowchart:

1. **Start:**
    - Program execution begins.
2. **Set Random Seed:**
    - Set a random seed using the current time to ensure different random numbers each time the program runs.
3. **Declare Variables:**
    - Declare a 9x9 array to represent the Sudoku puzzle.
4. **Initialize Variables:**
    - Display a welcome message.
    - Prompt the user to input the initial values for the Sudoku puzzle.
5. **Input Initial Sudoku Puzzle:**
    - Use nested loops to read the input values from the user and store them in the array.
    - Validate input: If the input is not between 0 and 9, display an error message and exit the program without failure.
6. **Display Initial Sudoku Puzzle:**
    - Display the input Sudoku puzzle with a visual representation of the 3x3 boxes.
7. **Solve Sudoku:**
    - Display a message indicating that the program is solving the Sudoku puzzle.
    - Use nested loops to fill in empty cells with random numbers between 1 and 9.
8. **Display Solved Sudoku Puzzle:**
    - Display the solved Sudoku puzzle with a visual representation of the 3x3 boxes.
9. **End:**
    - Program execution ends.