Sudoku Solver

We are given a 9×9 2D array 'grid [9][9]', the goal of this project is to assign digits (from 1 to 9) to the unassigned cells (marked as 0) so that every row, column, and subgrid of size 3×3 contains exactly one occurrence of the digits from 1 to 9.

Prerequisites

Visual Studio 2019 (with C++ Desktop Development workload)

Installation

1. Install and Download Visual Studio 2019 bootstrapper file

Check the system requirements to know whether your computer supports Visual Studio 2019. Visual Studio 2019 can be downloaded through the following link: *https://visualstudio.microsoft.com/downloads/*

2.Install the Visual Studio installer

From your Downloads folder, double-click the bootstrapper that matches or is similar to one of the following files:

- vs_community.exe for Visual Studio Community *(recommended)*
- vs_professional.exe for Visual Studio Professional
- vs_enterprise.exe for Visual Studio Enterprise

If you receive a User Account Control notice, choose 'Yes'.

3.Choose workloads

Find the workload you want in the Visual Studio Installer.

"Desktop development with C++" is a required workload for the project.

4.Start download and Install the Visual Studio 2019 setup

- Choose individual components (Optional)
- Install language packs (Optional)
- Select the installation location (Optional)
- Select "Install"

5.Start developing

After Visual Studio installation is complete, choose the Launch button to get started developing with Visual Studio.
On the start window, choose Create a new project.

Methods Used

1. **bool FindUnassignedLocation(int grid[N][N], int& row, int& col);**

Task: This function finds an entry in grid that is still unassigned.

Return Type: bool

Parameters: grid(sudoku), row, column

Working: Searches the grid to find an entry that is still unassigned. If found, the reference parameters row, col will be set the location that is unassigned, and true is returned. If no unassigned entries remain, false is returned.

2. **bool isSafe(int grid[N][N], int row, int col, int num);**

   Task: Checks whether it will be legal to assign number to the given row, column

   Return Type: bool

   Parameters: grid(sudoku), row, column, value (from 1 to 9) to be assigned

   Working: Returns a boolean which indicates whether it will be legal to assign num to the given row, col location.

3. **bool isValidSudoku(int board[][N]);**

   Task: Check if all elements of board [][] stores value in the range[1, 9].

   Return Type: bool

   Parameters: grid(sudoku)

   Working: the function *isValidSudoku* performs the following steps:

   - Declare *bool unique [N + 1];* Stores unique value from 1 to N.
   - *memset(unique, false, sizeof(unique));* will initialize unique[] array to false.
   - Initialize *int Z = board[i][j];* which will store the value of board[i][j]
   - Check if current row stores duplicate value.
   - Traverse each block of size 3 * 3 in board [][] array.
   - A variable (here 'j') stores first column of each 3 * 3 block.
   - Check if current block stores duplicate value.
   - *return true;* If all conditions are satisfied.

4. **bool SolveSudoku(int grid[N][N]);**

   Task: If there is no unassigned location, we are done else trigger backtracking.

   Return Type: bool

   Parameters: grid(sudoku)

   Working: Takes a partially filled-in grid and attempts to assign values to all unassigned locations in such a way to meet the requirements for Sudoku solution (non-duplication across rows, columns, and boxes).

5. **bool UsedInRow(int grid[N][N], int row, int num)**

   Task: returns true if grid [row][col] is equal to num else false.

   Return Type: bool

   Parameters: grid(sudoku), row

   Working: Returns a boolean which indicates whether an assigned entry in the specified row matches the given number.

6. **bool UsedInCol(int grid[N][N], int col, int num);**

   Task: returns true if grid [row][col] is equal to num else false.

   Return Type: bool

   Parameters: grid(sudoku), column

   Working: Returns a boolean which indicates whether an assigned entry in the specified column matches the given number.

7. **bool UsedInBox(int grid[N][N], int boxStartRow, int boxStartCol, int num)**

   Task: returns true if grid [row][col] is equal to num else false.

   Return Type: bool

   Parameters: grid(sudoku), column

   Working: Returns a boolean which indicates whether an assigned entry within the specified 3x3 box matches the given number.

8. **void printGrid(int grid[N][N]) ;**

   Task: prints sudoku grid

   Return Type: void

   Parameters: grid(sudoku)

   Working: A utility function to print grid by traversing through NxN rows and columns.

Driver Function

- Input a 9x9 array from the user such that all the unassigned spaces are input as 0.
- If the input is a partially filled grid (i.e. consists of ‘0’ input values) then the flag, f is assigned as 1 else it will remain 0.
- ***Case 1: if f=1,***
  Call function *SolveSudoku*, if it returns true print grid else print the message *“No solution exists “.*
- ***Case 2: if f=0,***
  Call function *isValidSudoku*, if it returns true print grid else print the message *“The given sudoku is complete but invalid”.*

Using Visual Studio 2019 to build and run the project

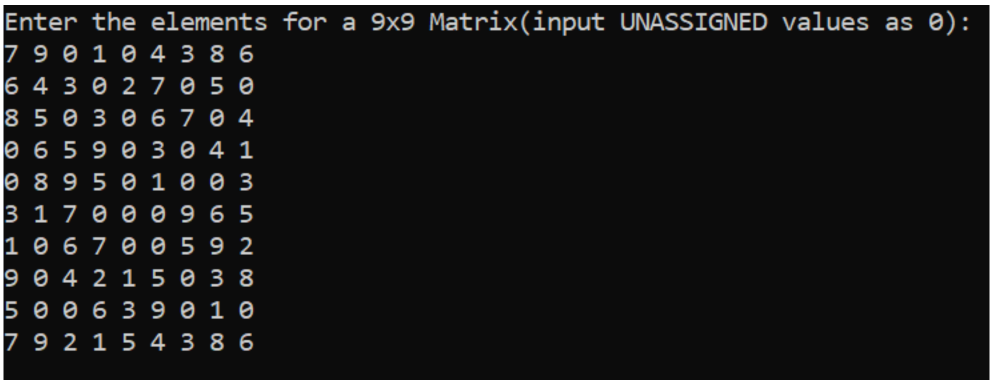
Select Build->Build Solution.
Select Debug->Start without debugging.

In the Terminal Window, input the values of sudoku replacing all the unassigned values by ‘0’ use whitespace or enter to separate the values.

A total of 81 values will be entered in order to solve a 9x9 Sudoku.

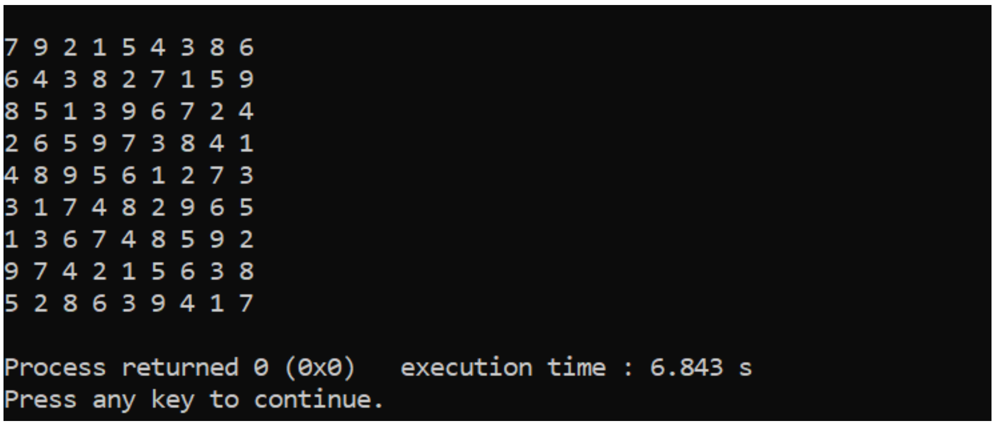
The Solved Sudoku will be printed as the output on press of any key.

Sample Input



```
Enter the elements for a 9x9 Matrix(input UNASSIGNED values as 0):
7 9 0 1 0 4 3 8 6
6 4 3 0 2 7 0 5 0
8 5 0 3 0 6 7 0 4
0 6 5 9 0 3 0 4 1
0 8 9 5 0 1 0 0 3
3 1 7 0 0 0 9 6 5
1 0 6 7 0 0 5 9 2
9 0 4 2 1 5 0 3 8
5 0 0 6 3 9 0 1 0
7 9 2 1 5 4 3 8 6
```

Sample Output



```
7 9 2 1 5 4 3 8 6
6 4 3 8 2 7 1 5 9
8 5 1 3 9 6 7 2 4
2 6 5 9 7 3 8 4 1
4 8 9 5 6 1 2 7 3
3 1 7 4 8 2 9 6 5
1 3 6 7 4 8 5 9 2
9 7 4 2 1 5 6 3 8
5 2 8 6 3 9 4 1 7

Process returned 0 (0x0)   execution time : 6.843 s
Press any key to continue.
```

Author

Name: Khushi Kumar

University: Graphic Era deemed to be University, Dehradun