2014-08-14 - GD2S03 - Summative_1_Lua - Alex Denford

Solver:

In my C++ code I check for a mouse click in the area of the "Solve" button which calls a function called "SolveSudoku". This function is also called if the user pushes 'S'. The SolveSudoku function calls the "Solve" function inside my LUA file "Sudoku Solder.lua". I pass in a string which contains the entire 2D array of the board. I found this to be the easiest way as passing a 2D array into LUA proved to be quite difficult.

The actual algorithm for solving is a form of brute force which uses "backtracking". This means that the function is recursive and it attempts to place a number at an empty spot, and then continues recusing until it either succeeds or it reaches a dead end. If the function reaches a dead end it undoes the last placed number and keeps trying. This recursive manner will always eventually find a solution if one is possible. A string is then returned from LUA to my C++ function which contains the resulting solution. If there is no solution the returned string is simply blank. This is checked and then a message box is displayed to the user.

Other key systems:

I chose to add keyboard inputs to allow for easy board setup. I have also added functionality so that the user can input setups via a file. These are in the form of a 9 x 9 grid of numbers.

I decided to extend the program so that it could also be used as an actually Sudoku playing system. For this I added a game start, ability to restart, clear and message boxes telling the user if they have completed the Sudoku.