CSC 330 – Artificial Intelligence
Program Assignment #4 – Logic and Sudoku
Due Friday, April 7, at 5:00 PM

Description:
The game of Sudoku is still relatively popular in this country. For those of you who don't know how to solve a Sudoku problem, here is a brief explanation of the rules:

- The game board consists of a 9x9 grid, further broken down into a 3x3 set of smaller 3x3 grids (see the example below) with each individual cell of the grid to eventually contain a digit between 1 and 9.
- The final goal is to fill the board with digits so that all of the following are true:
  - Each row of 9 cells contains the digits 1 through 9 in some order with no repeats or omissions.
  - Each column of 9 cells contains the digits 1 through 9 in some order with no repeats or omissions.
  - Each of the smaller 3x3 grids contains the digits 1 through 9 in some order with no repeats or omissions.

For example, here is an unfinished Sudoku puzzle:

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

As an example of how to solve something like this, consider the third column of the grid (the one containing a 6, 1, 8, 7, and 9). This column needs a 2, 3, 4, and 5 to complete it. Consider the number 3 for a minute. The top empty cell (between the 6 and 1) is in a row that already contains a 3, so the 3 cannot go there. The second and third cells (between the 8 and the 7) are in a 3x3 grid that already contains a 3, so the 3 cannot go in either of those cells. This leaves the bottom cell of that column, so the 3 must go in that cell. Using similar (and more complicated) logic, we can fill the rest of the squares in the grid.

The Program:

For this assignment, you will be writing a program to automatically solve Sodoku puzzles. Your program should do the following:

- Prompt the user to enter a file name.
- Open that file and read in the Sudoku puzzle contained within (the file will consist of 9 rows of 9 single digit numbers each, with zeroes representing the initially empty cells). Display the initial problem you have read in, as a 9x9 grid.
- The program should then solve the puzzle by filling in the empty cells of the grid. Display the final solution.

You may use whatever solution method you wish, **except** brute force (meaning you cannot simply consider every possible configuration that is possible until you find one that works). Brute force would likely not finish in a reasonable amount of time anyway. The easiest way should be to logically consider the possibilities for each cell, eliminating possibilities until only one is left for a cell. Once you have filled in some cells using logical reasoning, a search technique can be used to finish, although every puzzle should be able to be solved using only logical reasoning.

Some Notes:

- Note that you do not to use formal propositional or first-order logic to solve this problem. You should use simple logical reasoning.
- There are some sample Sudoku puzzles provided on the course Moodle site. Sudoku puzzles typically come with particular difficulty levels – either "easy, medium, hard, very hard" or a rating from 1 to 5 stars seem to be the most prevalent.
- Do not put this off until the last minute, and put some thought into how to implement the state of the puzzle.

Specifications:

1. Your program represents a sincere attempt to solve the problem.
2. Your program compiles and runs, assuming specification 1 is met.
3. Your program successfully loads and prints the puzzle from the file as a 9x9 grid.
4. Your program uses some form of logical reasoning to solve the puzzle.
5. Your program successfully solves the "easy" test puzzle provided on Moodle.
6. Your program successfully solves the "medium" test puzzle provided on Moodle.
7. Your program successfully solves the "hard" test puzzle provided on Moodle.
8. Your program successfully solves the "very hard" test puzzle provided on Moodle.
9. Your program successfully solves a 1-star level Sudoku puzzle of my choice.
10. Your program successfully solves a 2-star level Sudoku puzzle of my choice.
11. Your program successfully solves a 3-star level Sudoku puzzle of my choice.
12. Your program successfully solves a 4-star level Sudoku puzzle of my choice.
13. Your program successfully solves a 5-star level Sudoku puzzle of my choice.
14. Your program is well-written and styled (formatting, consistent indentation, comments, etc.).