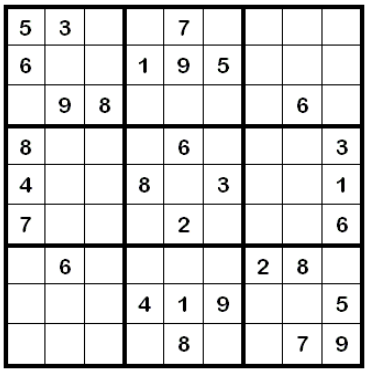
CS3270 Spring 2017
Programming Assignment No. 1
Due: Thursday, February 16, 11:59pm

Purpose: Gain experience in advanced C++ OOP programming and recursive backtracking algorithms (and possibly constraint programming).

Assignment: Create a **Sudoku** solver.

Sudoku is a popular puzzle where you fill in numbers on a grid, trying to keep certain conditions true. To learn more about how Sudoku works, check out http://en.wikipedia.org/wiki/Sudoku. You'll find a sample puzzle and an explanation of the rules.

Write a C++ class that reads a file containing an unfinished Sudoku puzzle, then solves the puzzle using the method(s) of your choice. A simple recursive backtracking algorithm will always work, but may not be the most efficient. A simple constraint solver may be more efficient and will handle easy puzzles, but would require additional help (i.e., a recursive routine) for more difficult problems. Once the puzzle has been solved, display the results to the screen. The input file will simply contain the numbers in the puzzle delimited by spaces on each line, where zeroes are used to indicate unknowns. For example, this puzzle:



(http://commons.wikimedia.org/wiki/Image:Sudoku-by-L2G-20050714.gif)

would be represented in your input as:

```
5 3 0 0 7 0 0 0 0
6 0 0 1 9 5 0 0 0
0 9 8 0 0 0 0 6 0
8 0 0 0 6 0 0 0 3
4 0 0 8 0 3 0 0 1
7 0 0 0 2 0 0 0 6
0 6 0 0 0 0 2 8 0
0 0 0 4 1 9 0 0 5
0 0 0 0 8 0 0 7 9
```

When your program prints out the solution, it should print it out to the screen in the following format:

```
5 3 4 | 6 7 8 | 9 1 2
6 7 2 | 1 9 5 | 3 4 8
1 9 8 | 3 4 2 | 5 6 7
------+-------+------
8 5 9 | 7 6 1 | 4 2 3
4 2 6 | 8 5 3 | 7 9 1
7 1 3 | 9 2 4 | 8 5 6
------+-------+------
9 6 1 | 5 3 7 | 2 8 4
2 8 7 | 4 1 9 | 6 3 5
3 4 5 | 2 8 6 | 1 7 9
```

Functional Specifications: You are to create a C++ class called `Sudoku`. This class should be implemented in the files "`Sudoku.h`" and "`Sudoku.cpp`". The public interface of this class is as follows:

`Sudoku()`	Default constructor. Should initialize the object with an empty puzzle (81 zeroes).
`void loadFromFile(string filename)`	Reinitializes the object with a new puzzle from the specified file. You may assume the input file has the specified format. Note: It is recommended that you use the extraction operator to read in the values, as it will automatically skip all white spaces (blanks, newlines) for you.
`bool solve()`	The entry point for your solver. Returns true if a solution was found, otherwise returns false. If no solution was found, you may leave the puzzle in either (1) its initial state, or (2) the state when you determined no solution was possible.
`void print() const`	Prints the current puzzle contents to the screen in a nicely formatted manner.
`bool equals(const Sudoku &other) const`	Determines if two puzzles are equal.

A driver program which uses this interface is being supplied to you. You will likely have to provide several private functions to actually solve the puzzle. You are to provide a solution method of your own design. Programming style will be a part of your grade so use good style and document your code with comments. Add block comments to the top of all files and include an academic honesty statement (as you have done in other CS courses).

All grading will be done using CLion/clang. Your code is expected to compile cleanly and run with that compiler.

For comparison purposes, you will time your solution program. We want this information so that we can compare our C++ code against solutions written in other programming languages. How fast your program finds a solution will ***not*** be a part of your grade – unless your solution is *grossly inefficient* [we will define "grossly inefficient" as being 5x slower than my simple-minded solver, or one that exhausts all stack space].

Deadlines: To make sure everyone is making sufficient progress on this project, we will have multiple deadlines. They are as follows (you should review the course's syllabus regarding late penalties):

2/13/17	Submission of code to date. The bulk of your code should be written (though not fully operational) at this time. Submit all your .h & .cpp files – do not submit a zip file of your entire project directory. It is expected that you have completed at least the following at this checkpoint: the constructor, the `loadFromFile()` method, the `print()` method, and the `equals()` method. Failure to have made progress to this point will cost you 10% of your final grade on the project.
2/16/17	Final project submission. Code should be fully commented. Submit all your .h & .cpp files – do not submit a zip file of your entire project directory.

Academic honesty: As stated in class, there are many solutions to Sudoku in many different programming languages available on the Internet. Do not look at the code you may find there. Using code that you find on the Internet is unethical, and of course you would miss the learning opportunity that you get by developing this yourselves. This instructor will report any violations to the university's Honor Council.