**CS6456 (F2015) Operating Systems**

**Project 3**

**Title: Sudoku Solution Validator**

**Due: September 28, 2015 (11:00 a.m.)**

**Points: 10**

Write a program in C/C++ to determine whether the solution to a Sudoku puzzle is valid (p. 197). Your program should be designed in such a way that it contains multiple threads working together to determine the answer.

(1) The project is not a team work.

(2) Your program will be compiled and run in the Debian VM environment.

(3) You should read the problem shown on pp. 197-199 carefully and understand thoroughly what the problem your program will solve.

(4) You should use multithreaded approach shown in class to design your program. You may follow the strategy suggested in the book or use a design methodology different from the one shown in the book. Regardless of what design methodology you are using, your program must contain at least three threads: one thread to check each row, one thread to check each column, and one thread to check each 3×3 subgrid.

(5) The numbers in the 9×9 grid are stored in a file named **sudoku.txt**. The numbers are stored in the file in a contiguous way, with each pair of numbers separated by exactly one space. Your program will read in all the numbers and store them in a two-dimensional array with 9 rows and 9 columns. You may assume that all the numbers stored in the file are of **int** type. However, each number in the grid can be either positive, 0, or negative.

(6) The output from your program will be one of the following. (i) If all the numbers in the 9×9 grid are between 1 and 9 (inclusive), print a message indicating that it is a valid sudoku. (ii) If some of the numbers in the 9×9 grid are out of the range from 1 to 9 (inclusive), print the row and the column that contain the number for each of them. Make sure that the row and the column are printed only ONCE for each number that is out of range.

(7) You should use a make file to compile your program.

(8) Turn your project in via **collab** in a **tar** file (created using the **tar** command) consisting of (i) all headers accessed in the source code file; (ii) source code file for project 3; (iii) a make file; and (4) a write-up showing your design and implementation of the project. The **tar** file should be named as follows: **p3**, followed by the first letter of your first name, followed by your last name, and followed by the file extension (i.e. **tar**). For example, the tar file turned in by **John Smith** should be named **p3jsmith.tar**.

(9) Submit also a hardcopy of the **tar** file in class consisting of the four items shown above.

(10) Your program should be successfully compiled. Programs failing to compile will not get any points.