Programming Assignment One

Finish all the tasks below. Submit the code for all questions, that is, submit three .c programs. Add the proper descriptions as the comments at the beginning of the code, including author, student ID, function description, and create date. Refer to Rubrics for the grading criteria.

This assignment will involve writing a multithreaded program to validate a Sudoku solution. Please refer to the programs in slides 27-29 in the lecture Chapter 4 (ch4_Threads.pdf) to do this assignment and run the program on a Linux system.

Sudoku

A Sudoku puzzle uses a 9 × 9 grid in which each column and row, as well as each of the nine 3 × 3 sub-grids, must contain all of the digits 1 ⋯ 9. Figure 1 presents an example of a valid Sudoku puzzle.
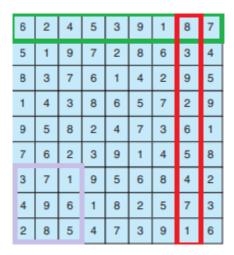

Fig. 1 Solution to a 9 x 9 Sudoku puzzle

Threads

This project consists of designing a multithreaded application that determines whether the solution to a Sudoku puzzle is valid. There are several different ways of multithreading this application. One suggested strategy is to create threads that check the following criteria:
1. A thread to check that each column contains the digits 1 through 9
2. A thread to check that each row contains the digits 1 through 9
3. Nine threads to check that each of the 3 × 3 sub-grids contains the digits 1 through 9

This would result in a total of 11 separate threads for validating a Sudoku puzzle. However, you are welcome to create even more threads for this project. For example, rather than creating one thread that checks all nine columns, you could create nine separate threads and have each of them check one column.

Passing Parameters to Each Thread

The parent thread will create the worker threads, passing each worker the location that it must check in the Sudoku grid. This step will require passing several parameters to each thread. The easiest approach is to create a data structure using a **struct**. For example, a structure to pass the row and column where a thread must begin validating would appear as

follows:

```
/* structure for passing data to threads */
typedef struct {
    int row;
    int column;
} parameters;
```

The Pthreads program will create worker threads using a strategy similar to that shown below:

```
parameters *data = (parameters *)malloc(sizeof(parameters));
data->row = 1;
data->column = 1;
/* Now create the thread passing it data as a parameter */
```

The data pointer will be passed to the pthread_create() (Pthreads) function, which in turn will pass it as a parameter to the function that is to run as a separate thread.

Returning Results to the Parent

Each worker thread is assigned a task of determining the validity of a particular region of the Sudoku puzzle. Once a worker has performed this check, it must pass its results back to the parent. One good way to handle this is to create an array of integer values that is visible to each thread. The ith index in this array corresponds to the ith worker thread. If a worker sets its corresponding value to 1, it is indicating that its region of the Sudoku puzzle is valid. A value of 0 indicates otherwise. When all worker threads have completed, the parent thread checks each entry in the result array to determine if the Sudoku puzzle is valid.

You program should first read in a text file containing a possible sudoku solution data, then start a few threads to validate it. The data file looks like the following:

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

There are 9 rows, with 9 integers in each row, separated by a coma. You can copy the above data to a .txt file, then in your program use fscanf() function to read the data to an array. You may want to print out the data in the array to make sure the data is read correctly into your array.

Commands to Compile and Run Your Program

Suppose your code file name is sudoku.c (please refer to the submission part at the end of

this document for the naming convention）, use the following command to build an executable file Sudoku:

>gcc -o sudoku sudoku.c –lpthread

Use the following command to run the executable program (assume that sudoku_data.txt is the txt file that stores the Sudoku data)

>./sudoku sudoku_data.txt

Sample Picture of Running Result:

Case 1: We didn't provide data file, The code should display some message to notify the user.

```
@OS-CST:~/24S/ProgrammingAssignmentOne$ ./sudoku
      You need to provide a sudoku data file!
      Usage: ./sudoku <filename>
```

Case 2: Now we provide a data file

```
@OS-CST:~/24S/ProgrammingAssignmentOne$ ./sudoku sudoku_data.txt
      6       2       4       5       3       9       1       8       7
      5       1       9       7       2       8       6       3       4
      8       3       7       6       1       4       2       9       5
      1       4       3       8       6       5       7       2       9
      9       5       8       2       4       7       3       6       1
      7       6       2       3       9       1       4       5       8
      3       7       1       9       5       6       8       4       2
      4       9       6       1       8       2       5       7       3
      2       8       5       4       7       3       9       1       6

      This sudoku solution is valid
```

Case 3：We provide a wrong data, it shows it is invalid.

```
@OS-CST:~/24S/ProgrammingAssignmentOne$ ./sudoku sudoku_data2.txt
      6       2       4       5       3       9       1       8       7
      5       1       9       7       2       8       6       3       4
      8       3       7       6       1       4       2       9       5
      1       4       3       8       6       5       7       2       9
      9       5       8       2       4       7       3       6       1
      7       6       2       3       9       1       4       5       8
      3       7       1       9       5       6       8       4       2
      4       9       6       1       8       2       8       7       3
      2       8       5       4       7       3       9       1       6

      Invalid sudoku solution
```

Submission

Submit only .c file into iSpace before the deadline. The file name is in the format A1_####.c where #### are last four digits of your student ID.