CPSC 380

Joseph Hansen
jhansen@chapman.edu

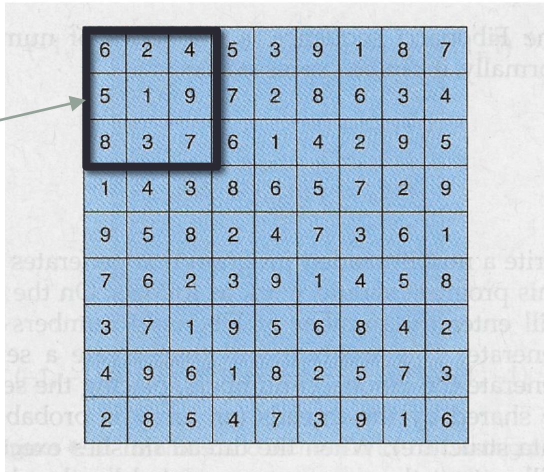
PROJECT 1

Project Description

Sudoku Solution Validator

- Game rules
  - A Sudoku puzzle uses a 9x9 grid in which each column and row, as well as each of the nine 3x3 sub-grids, must contain contain all of the digits 1…9. The figure below presents an example of a valid puzzle. This project consists of designing a multi-threaded application that determines whether the solution is valid, and if not, must recommend how to fix it.

Sub-grid



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

Project Requirements

- The Sudoku Validator Program (SVP) shall validate a 9x9 completed puzzle for correctness
  - Correctness means: each column and row must contain all of the digits 1-9 once
  - Each of the nine 3x3 sub-grids must contain all of the digits 1-9 once
- The SVP shall accept as input an ASCII .txt file with the elements separated by commas and structured as a 9x9 table
  - Sample valid test case provided for testing
  - Failed test cases will be provided and need to be demonstrated as part of the project
- The SVP shall specify via output to the user where the error occurred (e.g. row 3 column 5 – **preference would be 1-based answers**)
- Once the SVP detects an error, it must also specify what is the correct solution for that cell
- The SVP shall provide user feedback to the GUI or CLI to help explain what is going on and the final answer
  - Printouts of steps as they progress may be helpful

Project Implementation

- The SVP must contain at least 3 threads not-including the main thread (max thread limit shall not exceed 10 including main)
  - Final answer must be provided by main thread after the helper threads complete their job
    - Helper threads can provide intermediate feedback if a failure is detected
- Programming language and O/S are optional (language limited to C/C++/JAVA)
  - Preference would be to use POSIX threads on Linux, but others are okay if necessary
  - Explicit threads MUST be used – not implicit
    - Must demonstrate creation of threads, exit of threads and joining of helper threads with main one

Project Artifacts

- Students **must** turn in the following:
  - Source code
  - Executable
  - Output of the program (screen capture or pipe results to an output file) for each test case provided
  - A brief design description that includes how many threads were used and why. A block diagram would help too. Limit to 1 page max
- Project will be presented to professor and TA at a designated appointment time