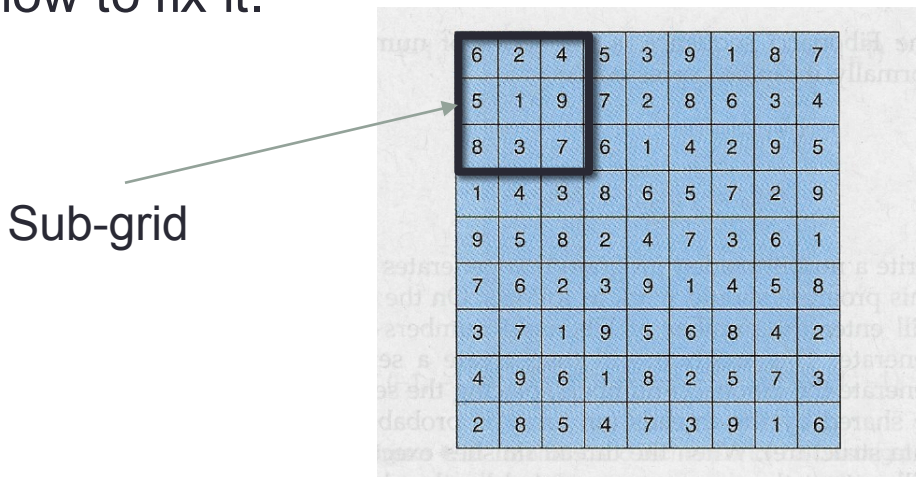
PROJECT 1

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.



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
  - Several valid and invalid test cases will be provided to test the software
- The SVP shall specify via output to the user where the error occurred (e.g. row 3 column 5 – **answers should be 1-based**)
- 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 the program executes may be helpful – but not required

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 the 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)
  - Explicit threads MUST be used – not implicit
    - Must demonstrate creation of threads, exit of threads and joining of helper threads with main one
- When determining if there is an error in a row, column or sub-grid, do NOT use the sum total method, since you may have two errors that could provide a correct sum (e.g. two wrongs make a right)

Project Artifacts

- Students **must** turn in the following:
  - Source code
  - 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