SSV

Sudoku Solution Validator

Problem

To design a multithreaded application that determines whether the solution to a Sudoku puzzle is valid.

- Passing parameters to each thread
- Returning results to the parent thread

Solution

The solution is implemented in C using pthreads. The program reads a Sudoku solution generated by a predifined function and checks the validity of the solution by creating 3 threads to check the rows, columns, and subgrids of the Sudoku solution. The program uses the following functions:

- `createRandomPuzzle`: A function that generates a random Sudoku solution. In common sense, a random Sudoku solution is not guaranteed to be valid.
- `createValidSudoku`: A function that creates a valid Sudoku solution.
- `displaySudoku`: A function that displays the Sudoku solution.
- `checkRow`: A function that checks the validity of each row in the Sudoku solution.
- `checkCol`: A function that checks the validity of each column in the Sudoku solution.
- `checkBox`: A function that checks the validity of each 3x3 subgrid in the Sudoku solution.
- `checkSudoku`: A function that creates 3 threads to check the validity of rows, columns, and subgrids in the Sudoku solution.

Note that the functions `checkRow`, `checkCol`, and `checkBox` are called by the threads to check the validity of the Sudoku solution. In order to address the dissapointment of an invalid solution, the program dereferences a null pointer to cause a segmentation fault, terminating the program.

In this way, player should never fail to complete a valid Sudoku solution, as the program will crash. It is advisable to include this feature in the difficulty level of Asian of Sudoku games.

Instructions

To compile and run the program, use the following commands:

```
make
```

user will be prompted to choose between a random or a valid Sudoku solution where 0 indicates a valid solution and 1 indicates a random solution. The program will then display the Sudoku solution and check its validity.