Assignment P4

Due Date: March 31

Purpose
In this assignment, you will use a two-dimensional array to determine if a given set of numbers is a valid Sudoku solution. Along the way, you will continue to practice programming with functions.

Problem
Sudoku fever is still running high in OOP Nation, almost as high as pickleball. Everywhere you turn, people are scribbling numbers 1–9 onto a 9×9 grid. Your program will have two main functions: 1) reading in a puzzle and displaying it as a nicely formatted Sudoku grid, and 2) checking if the solution is correct.

Sudoku
The game is simple (the program, perhaps, not so much). The player starts with a 9×9 grid, with some cells containing numbers. The player has to fill in the remaining grid boxes such that every row, every column, and every 3×3 box contains the digits 1–9. Below is a finished puzzle; the darker numbers represent the starting puzzle:

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

To see a more thorough explanation of the rules, go to: https://www.sudoku.name/rules/en. To play online, go to: https://www.websudoku.com/.

Input
The Sudoku solution is stored in a text (.txt) file. Your program should prompt for the name of this file. The file contains 9 rows of 9 numbers. For example, a valid input file looks like the following:

```
123456789
234567891
345678912
456789123
567891234
678912345
789123456
891234567
912345687
```

The program must read and store all of the data in this file, a sample of which is available on the course web page. This should be stored in a two-dimensional array.

Output
You can get aim for a different number of points, depending on the effort you wish to put into the program. Follow these steps **in order**:

1. To get up to 70 points, the program should read the file and display the Sudoku board in a neat, aligned way. The output should include vertical and horizontal lines. For example, the file above should be displayed as follows (including spacing):

```
+-------+-------+-------+
| 1 2 3 | 4 5 6 | 7 8 9 |
| 2 3 4 | 5 6 7 | 8 9 1 |
| 3 4 5 | 6 7 8 | 9 1 2 |
+-------+-------+-------+
| 4 5 6 | 7 8 9 | 1 2 3 |
| 5 6 7 | 8 9 1 | 2 3 4 |
| 6 7 8 | 9 1 2 | 3 4 5 |
+-------+-------+-------+
| 7 8 9 | 1 2 3 | 4 5 6 |
| 8 9 1 | 2 3 4 | 5 6 7 |
| 9 1 2 | 3 4 5 | 6 8 7 |
+-------+-------+-------+
```

2. To get up to 85 points, the program must display the board as shown above. Additionally, it should determine if all the rows/columns are correct. Every row and column must include all the values from 1 to 9. The program should indicate if the puzzle is correct or if there is a problem in a row/column. The puzzle above is incorrect, because there are two 7s in the last column as well as two 8s in the 8^{th} column.

   If there is an error, be sure to indicate that there is a problem, AND indicate which row/column the error occurs.

   If there are no errors, then a message indicating that should be prominently displayed.

3. To get up to 100 points, the program must have all of the functionality described above. In addition, it should determine if each 3×3 box of numbers contains every value from 1 to 9 (while still following the row/column rules above). The Sudoku board above is hopelessly incorrect in this regard.

   If there is an error, be sure to indicate that there is a problem in a 3×3 box. Also indicate in which square the problem occurs; for this purpose, the squares are labeled as shown below:

```
+---+---+---+
| 1 | 2 | 3 |
+---+---+---+
| 4 | 5 | 6 |
+---+---+---+
| 7 | 8 | 9 |
+---+---+---+
```

IMPORTANT: Please indicate, as the first item in your output, up to which step you have completed. For example, if your program displays the board and determines if rows/columns are correct, display a message at the start of your output similar to:
`This program is complete through step 2.`

Specifics

- A 2D array should store the values in the Sudoku grid.
- The 2D array should be declared in `main()`. However, **no** cell should be accessed in this function (see next item).

- The `main()` function should prompt for the file name and then call other functions to do all of the processing. This includes individual functions to:
  - read the text file
  - check if a row/column is valid
  - check if a square is valid
  - display the board

  Additional functions are certainly allowed.
- All of your functions should be no longer than 25 lines. This is to encourage breaking the problem down into smaller subproblems.
- Remember to properly comment all of your code, including the functions and all parameters.
- Use appropriate parameters (pass-by-reference/value); do not pass arguments unnecessarily!

Notes

Don't try to solve every problem at once. This is why you should use multiple functions; each one will solve a smaller subproblem.

Have a plan for how to solve a subproblem. Only then should you write the function to carry out your plan.

Submit your source code in the usual way, using the normal naming convention.

I got distracted by learning.
– Amy Hopkinson '09