CS 100 Programming I
Project 2

Revision Date: October 2, 2015

Preamble

You may develop your code anywhere, but you must ensure it runs correctly on a Linux distribution before submission.

Sudoku!

> "Thanks for using up all my spare time!"
> – *Funny quotes from Sudoku players around the world at Sudoku.ca*

Your task is to program the fun and challenging game of Sudoku. You can read more about the game of Sudoku on the Web.

Sudoku rules

The rules for Sudoku are as follows. The goal is to fill a 9x9 grid such that each row and each column of the grid contain all of the digits from 1 to 9. In addition, each of the nine 3x3 subgrids that comprise the grid must also contain all of the digits from 1 to 9. A puzzle setter is used to place some initial numbers in the grid, such that the puzzle has a unique solution.

Checking for a valid solution

Your program must check for a correct solution in the following order:

1. Check each row from row zero to row eight.
2. Check each column from column zero to row eight.
3. Check each grid from upper left to lower right, row-ordered.

Once an error is encountered, there is no reason to continue checking the grid solution. Instead, the checking stops and an appropriate message is printed.

Data structures

You will store the putative solution in a two-dimensional square array. See Chapter 18 of *The Art and Craft of Programming* for information on creating a two-dimensional array.

Inputing values

The input file must have the following format. Each row in input the file will contain 3 values representing one value in the solution grid. The first number on each line in the file specifies a row in the grid, the second number specifies a column in the grid and the third value is the digit stored at that row and column. For example, consider the following lines:

```
8 0 3
3 4 6
```

The first line indicates the cell in row 8, column 0 of the solution grid contains the value 3 and the second line indicates the cell in row 3, column 4 of the solution grid contains the value 6. As indicated by this example, there is no particular order to the input. This file is free format. For example, you may place multiple or partial triplets on a line:

```
8 0
3 3 4 6
```

If a cell in the array does receive a value, you should place a zero at that location. Thus, the example above would be considered a valid input.

Output

Your program should first print the puzzle solution read in from the input file, as a two-dimensional array, as well as one of four messages, as appropriate:

1. The solution is correct.
2. There is an error in row *N*.
3. There is an error in column *N*.
4. There is an error in the subgrid *N,N*.

The wording does not have to be verbatim. *N* is zero based. For grids, *N,N* represents the row and column number of the upper left hand value in the 3x3 grid.

Reading from a file

To read from a free-format file, you may use the *Scanner* class, which you can retrieve with:

```
wget http://troll.cs.ua.edu/cs100/python/projects/scanner.py
```

Stepwise refinement

You should use stepwise refinement in implementing this project.

Level 0 Write a program that prints out the name of the file given as a command-line parameter that contains the Sudoku solution.

Level 1 Add code to the program to create an empty two-dimensional list with 9 rows and 9 columns, and prints out each line in this grid. Note this grid does not have any numerical values placed in it yet. Assuming you use the code given above, it will print out a "None" for each empty cell.

Level 2 Add code to the program that reads each line in the input file and places the specified value into its proper place in the two-dimensional list. Print out the two-dimensional list (the grid) in a pleasing format when finished.

Level 3 Add code to the program that tests each row of the grid for a correct Sudoku solution.

Level 4 Add code to the program that tests each column of the grid for a correct Sudoku solution.

Level 5 Add code to the program that tests each subgrid of the grid for a correct Sudoku solution.

After you complete each level above, you should name your program appropriately, *level0.py*, *level1.py*, *etc.* When you are ready to move to the next level, copy the previous level's program. For example:

```
cp level0.py level1.py
```

copies the code in *level0.py* and creates a new file named *level1.py*. Instructors and mentoring students have been informed that before they can help you, the name of your program must demonstrate what levels you have completed.

Program Organization

To receive full credit for this project, your code must be succinct. For example, this means you need to use loops and nested loops, where appropriate, instead of repeating the same code 9 times. Your program will be penalized severely if it contains repetitive code.

Name your final file *sudoku.py*. Make sure you have *scanner.py* in your directory as well.

Compliance Instructions

To make sure that you have implemented your program correctly, create two test files *test0* and *test1*.

Running your program on *test0* should look like this:

```
$ python3 sudoku.py test0

                column
      0  1  2   3  4  5   6  7  8
    +---------+---------+---------+
0 | 1  7  2 | 5  4  9 | 6  8  3 |
1 | 6  4  5 | 8  7  3 | 2  1  9 |
2 | 3  8  9 | 2  6  1 | 7  4  5 |
```

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

This Sudoku solution is correct!
$
```

Running your program on *test1* should look like this:

```
$ python3 sudoku.py test1
...
This Sudoku solution is incorrect!
There is an error in column 5
$
```

The actual solution has been elided in the above example, but should be present.

Adjust the contents of your test files until you achieve the desired output.

If your code does not produce the correct output or fails with a runtime error while running these two tests, then you will receive a zero for this assignment.

Challenge

If you are so inspired, you may wish to write a program to solve a sudoku puzzle: Sudoku Solver

Submission Instructions

Change to the directory containing your assignment. Then, run the command:

```
submit cs100 xxxx project2
```

Replace *xxxx* with your instructor name.