Sudoku Solver
v1.2
Cpt S 489

Read all instructions carefully before writing any code. Do all work individually!

The zip that contains these instructions also includes an HTML file. The HTML page is set up to reference the Sudoku9x9.js file from the previous assignment, and use it to run tests. If you change the HTML file to add extra test cases, make sure you test with the original HTML file as well to make sure everything works. You will submit only your JavaScript file for this assignment.

In this assignment, you will extend the Sudoku foundation classes to make a puzzle solver. There's only 1 function that needs to be added, although you can add more as needed to help this function work.

Add the **solve** function to the Sudoku9x9 class. This function does not take any parameters. The function should solve for all cells in the puzzle. Return an object with a **solved** member, which is a bool value. True indicates that the puzzle was fully solved, false indicates that it was not.

The HTML page includes several sample puzzles of varying difficulty taken from a variety of sources. You can look through the code to see how these sample puzzles are declared and add extras if you wish. But the included set should provide a good overall test of your code.

When you click the button on the page to solve the puzzle, your Sudoku9x9Obj.solve() call is timed. On a 3.5GHZ machine, you must be able to solve the puzzle in under 1000 milliseconds (1 second). To achieve this, you must implement many of the Sudoku logic solving rules. A brute-force approach will not suffice. That being said, implementing many logic rules just to narrow down possibilities and then doing a guess-and-check for the remainder may suffice for achieving <1 second solving time. It's up to you to find a balance and implement the solving logic so that it correctly solves the puzzle in the required time.
For a reference, my solver on a 3.5 GHZ machine solved the hardest puzzle in about 115 milliseconds. It took about 430 milliseconds on an iPhone 6. It also finds the exact number of possible solutions. I have verified that all sample puzzles only have 1 possible solution, and your solver only needs to solve puzzles with 1 possible solution. It's not a true Sudoku puzzle if it has more than 1 solution.

Scoring:
3/3 if all puzzles can be solved, each in less than 1000 milliseconds

2/3 if all puzzles with 75% difficulty or lower can be solved in under 1000 milliseconds. This means all puzzles that are:

- rated 3 or lower on a scale out of 4, or
- rated 4 or lower on a scale out of 6, or
- "Easy" or "Medium" on an Easy/Medium/Hard scale

1/3 if all puzzles with 50% difficulty or lower can be solved in under 1000 milliseconds.

0/3 for anything else