DM 503 Programming B

Fall 2010 Project (Part 1)

Department of Mathematics and Computer Science
University of Southern Denmark

November 18, 2010

Introduction

The purpose of the project for DM503 is to try in practice the use of recursion and object-oriented program design. The project consists of two parts.

Please make sure to read this entire note before starting your work on this part of the project. Pay close attention to the sections on deadlines, deliverables, and exam rules.

Exam Rules

This first part of the project is a part of the final exam. Both parts of the project have to be passed to pass the overall project.

Thus, the project must be done individually, and no cooperation is allowed beyond what is explicitly stated in this document.

Deliverables

- A short project report in PDF format (at least 3 pages without front page and appendix) has to be delivered. This report has to contain the following 7 sections (for details see the slides for September 14):
  - front page
  - specification
  - design
  - implementation
  - testing
  - conclusion
  - appendix
- All source code.

The deliverables have to be delivered using Blackboard's Assignment Hand-In functionality. Delivering by e-mail or to the teacher is only considered acceptable in case Blackboard is down before the deadline.

Deadline

December 2, 12:00

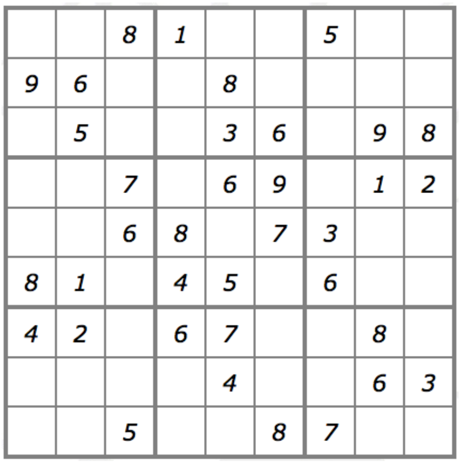
The Problem

Your task in this part of the project is to write a solver for Sudoku puzzles. Sudoku puzzles are a kind of crossword puzzles with numbers where the following two conditions have to be met:

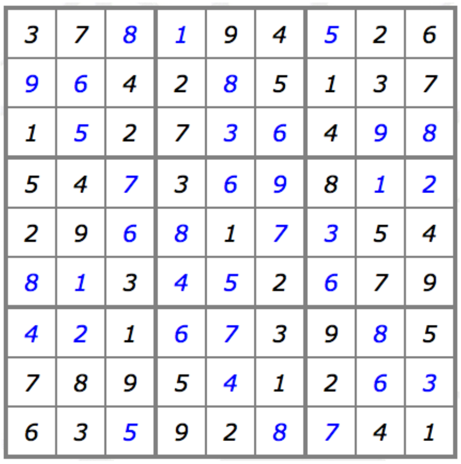
- In each row or column, the nine numbers have to be from the set $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ and they must all be different.

- For each of the nine non-overlapping 3x3 blocks, the nine numbers have to be from the set $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ and they must all be different.

The following two figures show a Sudoku puzzle and its solution.



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



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

The Input

For input to your program, the Sudoku puzzles are represented by text files. More specifically, they are represented as nine lines of nine characters separated by spaces. The characters can be of two types:

- A letter `X` signifies an empty cell, i.e., a cell that your solver needs to find a number for.

- A number signifies a cell whose value is fixed, i.e., the corresponding cell needs to have this number in the solution that you find.

Thus, for our example above we obtain the following text file:

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

The home page of the course contains a number of possible inputs to test your program on.

The Output

The output of your solver should be a representation of the Sudoku playing field for the solved puzzle. For our example above the output should thus be:

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

The Abstract Data Type

The Sudoku playing field can be modelled by an abstract data type based on arrays. It should implement the following operations:

- `public void fromFile(String fileName)` which intializes the playing field from an input file.
- `public boolean tryValue(int val, int i, int j)` which returns false if the value val cannot be placed at row i and column j and returns true and sets the cell to val otherwise.
- `public boolean isEmpty(int i, int j)` which checks if the cell at row i and column j is empty.
- `public void clear(int i, int j)` which sets the cell at row i and column j to empty.

A template for the abstract data type is available from the course home page as `Field.java`.

The Task

Write a solver for Sudoku puzzles that takes as argument the name of a file containing a Sudoku puzzle and prints the solution to the screen. Implement the solver USING RECURSION!

A template for the solver is available from the course home page as `Sudoku.java` and `SolvedException.java`.