Prifysgol Cymru
Aberystwyth
The University of Wales

Department of Computer Science

CS21120: Data Structures and Algorithm Analysis
Assignment 1 - **Sudoku**

1 Background

For this assignment you are required to do three things:

1. Design an algorithm to solve a Sudoku puzzle.
2. Design the data structures that will enable your algorithm to work.
3. Produce well a structured and commented java program, with evidence of testing.

2 The tasks in detail

2.1 Design an algorithm to solve Sudoku puzzles.

Sudoku puzzles are quite popular, they consist of a 9x9 grid of numbers, which in turn is subdivided into nine 3x3 grids.

The rules are very simple, "Fill in the grid so that every row, every column, and every 3x3 box contains the digits 1 through 9."

An example Sudoku puzzle:

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

You can find out more information at `http://www.sudoku.org.uk/`

2.2 Design an algorithm to solve puzzles

You should design an algorithm to solve the Sudoku puzzles. These puzzles can be solved without needing to brute-force them, there are several techniques that you can use to do this shown on `http://www.cluemaster.com/pages.php?id=4`. You might want to start by keeping a list of possible candidates for each square and then eliminating those numbers that occur on the same row, column or grid square.

Your algorithm to solve the puzzles must be presented as a flowchart, pseudo code or some other recognized method of describing an algorithm.

2.3 Design the data structures that will enable your algorithm to work.

Produce a design for your data that will allow you to store the grid and manipulate it in ways that are needed for your algorithm. If you need to address a 3x3 sub grid in the 9x9 main grid then there must be a simple way of addressing that subgrid.

Your design should be expressed as a set of UML class diagrams with supporting documentation.

You should provide written justification for the design decisions that you have taken, and explain how your classes work together.

2.4 Produce well a structured and commented java program, with evidence of testing.

Your program should read the data files provided on the course web site, and produce solutions to them in a similar form.

The data files are very simple, they consist of 9 rows, each of which has 9 characters on a row. If a square is unknown, the character will be a space, otherwise it will be the value contained within the square.

Your program does not have to have a Graphical User Interface, but for maximum marks you should provide one. Marks for the GUI will be a maximum of 10% of the marks for the whole assignment. The program may either just display the problem and the solution, or for bonus marks it may show the algorithm progressing towards a solution. If you choose this method, then you will need to investigate putting the solver into a separate thread. You could use a SwingWorker thread (`http://java.sun.com/products/jfc/tsc/articles/threads/threads2.html`). If you do read this article, please note that SwingWorker has become part of the core API now (`http://download.oracle.com/javase/7/docs/api/`), so you don't need to download the code and include it yourself. If you use any code, then make sure that you credit the original source, and annotate where you have made alterations.

3 The Submission

- You must submit a neatly hand written or typed description of your solver algorithm, this could be expressed as a flowchart, pseudo-code or another well recognized documentation method.
- You must submit a design for your data structures using UML diagrams where necessary. This should also include the justification for design decisions that you have made.
- You must submit a printed copy of all your Java source code for all new and modified classes, which should be well structured and commented.
- You must submit evidence of testing demonstrating how far your solver gets for at least three puzzles.
- You must also submit an electronic copy of your assignment including all documents as PDFs and code (source and class files) as a zip file to blackboard.

4 The Marking Scheme

This assignment is worth 25% of the marks for the course CS21120, therefore you are expected to spend somewhere around 30 hours working on it. Note also that the majority of the marks will be for getting the solver side of the assignment working - if you only have a nice GUI you will not get very many marks.

Algorithm design 25%

Data structure design 25%

Java code and testing 50%

5 Hand-in dates and times

This assignment should be posted into the Assignment post box located in reception between 10am and 4pm on Friday 25th October 2013. The electronic submission deadline is the same time.