Project 2 — Due, Oct 9th 2020
100 points

Sudoku is a number placement puzzle. In this puzzle you are given an N x N grid of cells. The grid itself is composed of M x K sub-grids. You can place a number, drawn from 1 to N, in any cell. Initially the grid will have some of its cells partially filled. The objective of the puzzle is to complete the grid so that:

1. Every cell contains a number.
2. No number appears twice in any row, column of the N x N grid or in any row, column of any of the M x K sub-grid.

Figure 1 (a) below is a 12 x 12 Sudoku puzzle made up of 3 x 4 sub-grids and Figure 1 (b) is the solution to this puzzle.

	11								4		
7			2	6			3	5		11	
	6	9		1	12			7		10	
	4	1					10	8		6	
	8		9				12	10			
2				11		1			9		
		8			2		4				7
			3	5				12		4	
	7		4	12					6	8	
	5		8			10	7		11	1	
	1		11	3			5	2			6
		3								12	

Figure 1(a)

8	11	12	1	7	10	5	2	6	4	3	9
7	10	4	2	6	8	9	3	5	12	11	1
3	6	9	5	1	12	4	11	7	2	10	8
11	4	1	12	9	5	2	10	8	7	6	3
6	8	5	9	4	3	7	12	10	1	2	11
2	3	7	10	11	6	1	8	4	9	5	12
5	12	8	6	10	2	11	4	1	3	9	7
1	9	11	3	5	7	8	6	12	10	4	2
10	7	2	4	12	1	3	9	11	6	8	5
12	5	6	8	2	9	10	7	3	11	1	4
9	1	10	11	3	4	12	5	2	8	7	6
4	2	3	7	8	11	6	1	9	5	12	10

Figure 1(b)

Your assignment is to write a program to solve the general Sudoku puzzle using finite domain CSP methods discussed in the class.

The input to your program will be a text file formatted as follows:

1. The first line will have three integers that fix the values for N, M and K in that order.
2. The next N lines describe the board configuration – one per row. An empty cell will be denoted by _.

So N, M, K and the rows in Figure 1(a) will be represented like this:

12, 3, 4
_,11,_,_,_,_,_,_,_,4,_,_
7,_,_,2,6,_,_,3,5,_,11,_
.
.
.
_,_,3_,_,_,_,_,_,_,12,_

(Input format)

You should implement

1. Backtracking + MRV heuristic
2. Backtracking + MRV + Forward Checking
3. Backtracking + MRV + Constraint Propagation

Given an input instance you will generate an output for each of the 3 implementations. Each output will have the solution to the puzzle and the statistics about the solution. The first N lines generated by the output should describe the solved puzzle with the format introduced before. The last line should be the number of consistency checks done to arrive at the solution for each implementation.

You can find random sudoku examples from the web to test your program. Based on your test file analyze the pros and cons of these 3 implementations and write your findings in a report. Make sure that your input file follows the input format.

Notes:

1. Your program should be written in Python3
2. There are numerous sources on the Web for the Sudoku puzzle. You are encouraged to consult them to gain a good understanding of the puzzle.

Submission:

Submit by 11:59 pm on due date via Black Board. Put all the documents in a folder and zip it. The file name should be LastName_FirstName_HW2.zip.

- Source code with good documentation. Make sure it compiles.
- A trace of the execution for each strategy
- A report with the required statistics generated and any findings you discovered.