Pr. Mohamed RHAZZAF

ECOLE MOHAMMADIA D'INGÉNIEURS

Hands-On Exercises 1

Building a Sudoku Solver in Python

Objectives

Students will design, implement, and test a Sudoku-solving algorithm using constraint reasoning.

Language: Python (numpy package)

Environment: jupyter notebook or google colab

Introduction

Sudoku is a 9×9 logic puzzle requiring each row, column, and 3×3 subgrid to contain the digits 1–9 without repetition. Students will build a step-by-step solver using Python, focusing on grid representation, constraint extraction, candidate evaluation.

1- Create the Sudoku Environment

Task:

- Create a 9×9 matrix using lists or NumPy.
- Use 0 to represent empty cells.
- Print the board cleanly.

2- Identify Row, Column, and Cell permitted values

- Implement functions get_row(board,r), get_col(board,c), and get_cell(board,r,c) that return the permitted values for each structure.
- The 3×3 subgrid is determined by (r//3, c//3).

3- Candidate Extraction

Task:

- Write get_candidates(board, r, c) to get the permitted values for a given coordinate r and c
- Must return list of digits 1–9 not present in row, column, or cell.

4- Sudoku Solver via uncontrolled loops

Task:

- Write solve(board) using while loop (until you find all the candidate).
  - For all the empty cell get candidates, and if you have just one update the board
  - Otherwise, you will keep moving to the next empty cell

5- Test our Solver

Solve the following boards

Pr. Mohamed RHAZZAF

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

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

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

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

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

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