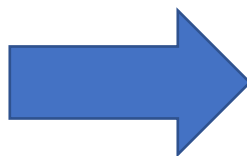# Individual Project 1

- **Deliverable:** Solving the Sudoku puzzle using Depth First Search (DFS). You will submit a java file (in plain text) which implements an interface to solve it.

- Due date: **3/3/2022 (11:59pm)**

Sudoku Puzzle Rules

- A 9x9 grid of numbers must be filled in
- Each number from 1 to 9 must appear exactly once in each row, in each column, and in each 3 x 3 "box"

Solve a number puzzle

Game Rule: every row, every column and every box needs to contain each of the numbers from 1-9 **exactly once**

Solve a number puzzle

Column

Game Rule: every row, every column and every box needs to contain each of the numbers from 1-9 **exactly once**

Solve a number puzzle

Box

Game Rule: every row, every column and every box needs to contain each of the numbers from 1-9 **exactly once**

Solve a number puzzle

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

Game Rule: every row, every column and every box needs to contain each of the numbers from 1-9 **exactly once**

Solve a number puzzle

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

Solve a number puzzle

Solve a number puzzle

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

Solve a number puzzle

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

Solve a number puzzle

Solve a number puzzle

Solve a number puzzle

Solve a number puzzle

Solve a number puzzle

Solve a number puzzle

Solve a number puzzle

3 (4-9)	1 (2-9)	5 (6-9)	7	4 (5-9)	6 (7-9)	8	9	2 (3-9)
6 (7-9)		2			1	3		4
	7		2	8				
9	4				5			1
1	3	8				7	5	9
5			1				8	3
				7	2		4	
7		9	6			2		
	6	4			8			

Solve a number puzzle

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

Pencil marks (top row): 4-9, 2-9, 6-9, 5-9, 7-9, 3-9; (second row): 7-9, 9

Solve a number puzzle

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

Solve a number puzzle

Solve a number puzzle

Solve a number puzzle

Solve a number puzzle

Solve a number puzzle

Complexity of the solution

- Branching factor b=9
- Solution Depth m=81
- Complexity $O(b^m)$

Code Template

- SudokuSolver.java
- SudokuUtil.java
- SudokuRunner.java: main program to run the solver

Code Template

- SudokuSolver.java: an interface that defines the main methods you need to implement including

  - solve_dfs: use depth first search to solve the puzzle
  - check_move: to check whether the current move is legal

  Your main task is to implement this interface!
  You only need to submit this to the Blackboard as a plain text file !

  If you don't know what a java interface is, please check this tutorial:
      https://www.programiz.com/java-programming/interfaces

Code Template

- SudokuUtil.java: define the main utility functions that you can use to help load, check and print the puzzle

  - load_problem: to load a starting board configuration
  - Write_matrix: to output a board configuration
  - check_solution: to verify whether the solution is correct.

- You should not need to change this file

Code Template

- SudokuRunner.java: define how you can run the program in the main method
    - What you need to do:

    (a) Implement your own SudokuSolver_YourName class which implements the interface SudokuSolver.

    (b) In SudokuRunner.java, change the file path to where test input puzzles are. Change SudokuSolver_XXX to SudokuSolver_[your own name].

Code Template

Replace this with your name

```java
public class SudokuSolver_YourName implements SudokuSolver {

    public int[][] solve_dfs(int i, int j, int[][] cells) {
        // TODO
    }

    public boolean check_move(int i, int j, int val, int[][] cells) {
        // TODO
    }

}
```

Put this in a new file, SudokuSolver_YourName.java (replace YourName with your own name).

I recommend that you implement DFS as **backtracking search**, rather than explicitly implementing the frontier queue.
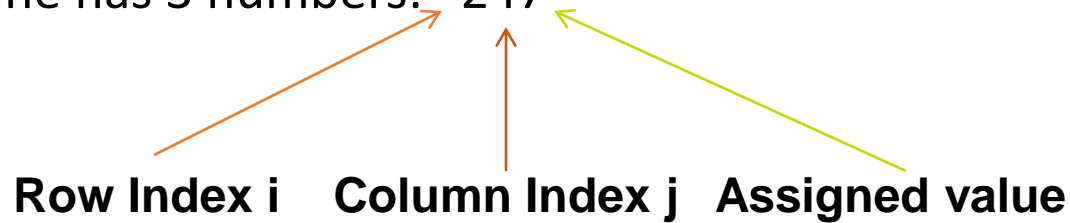
Sample Puzzles

- 3 puzzle samples will be provided
  - Test your solver using all of them
- Input Format:

  Each line has 3 numbers: 247

  Row Index i   Column Index j   Assigned value

- Assessment: correctness at solving the 3 sample puzzles, and one new puzzle (equal weight).