Alexandria University

Faculty of Engineering

CSED 2025

Paradigms Project Phase 2

Report

Submitted to

Dr. Mohamed M. Saad

Eng. Mohamed Tarek El Habiby

Eng. Menna El Kammah

Eng. Maram Aly Attia

Submitted by

Islam Yasser Mahmoud	**20010312**
Mohamed Ehab Mabrouk	**20011524**
Mohamed Hassan Youssef	**20011544**
Mkario Michel Azer	**20011982**

Faculty of engineering Alex. University

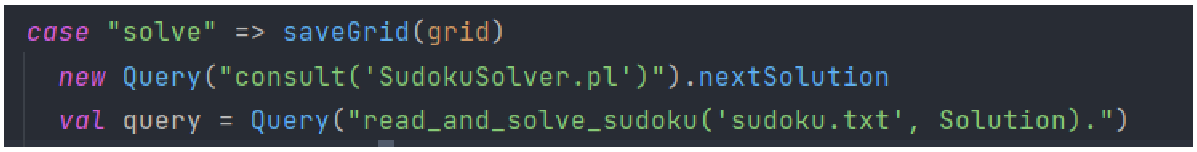
May 2023

Problem Statement:

investigate the features of the logical programming paradigm using Prolog by implementing a solver for the 2 single player games Sudoku and 8-Queens by solving the puzzle starting from a valid initial state or after the player has played some moves.

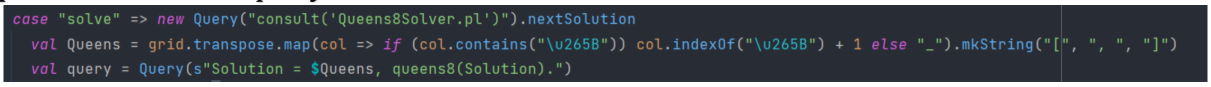
Solution:

- Two files **(SudokuSolver.pl)** & **(Queens8Solver.pl)** which contain the logic implementation of the solver of each game in Prolog
- The Prolog queries was written using **org.jpl7** library in Scala functional programming language
- In Sudoku the board is saved to a file then Prolog reads the file and solve the Sudoku puzzle based on it



```
case "solve" => saveGrid(grid)
  new Query("consult('SudokuSolver.pl')").nextSolution
  val query = Query("read_and_solve_sudoku('sudoku.txt', Solution).")
```

- In 8-Queens a string is constructed which describes the current 8-Queens board and is passed within the query

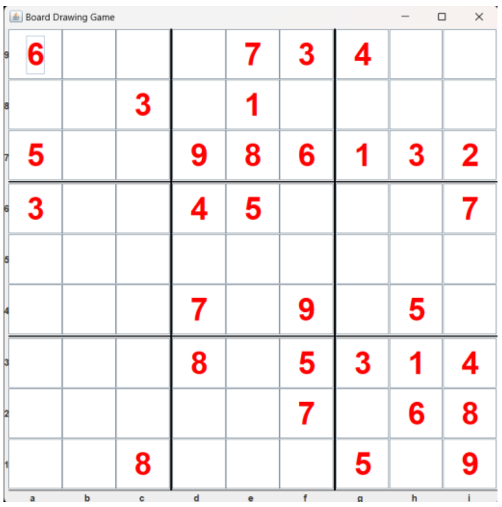


```
case "solve" => new Query("consult('Queens8Solver.pl')").nextSolution
  val Queens = grid.transpose.map(col => if (col.contains("\u265B")) col.indexOf("\u265B") + 1 else "_").mkString("[", ", ", "]")
  val query = Query(s"Solution = $Queens, queens8(Solution).")
```

- Any time the player wants to solve the puzzle while playing just type in the console "solve"
- If the current board in not solvable the Prolog captures this and so do the game engine which passes back to the Abstract game engine notSolvableFunction() which is then being applied in the Abstract game engine showing an appropriate error message to the user
- If the current board is solvable so the Prolog solves it then the game engine updates the board accordingly and passes it back to the Abstract game engine which calls the drawer with the updated solved board

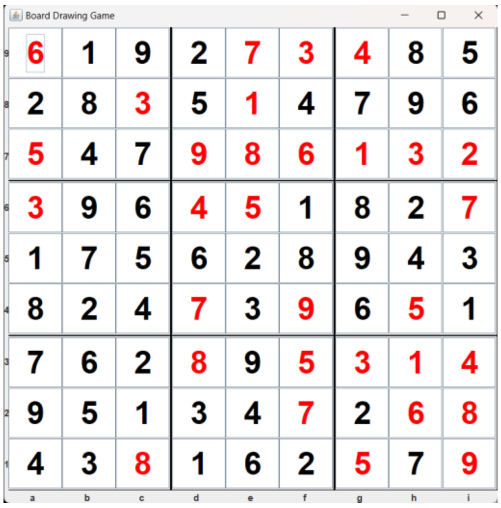
Test Cases:

- **Sudoku-Initial state** Red Numbers: Initial, **Black Numbers: played by user.**



Board Drawing Game

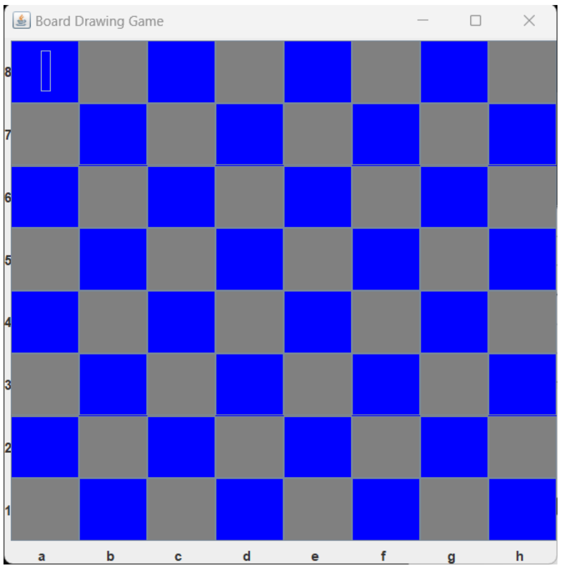
9	6				7	3	4		
8			3		1				
7	5			9	8	6	1	3	2
6	3			4	5				7
5									
4				7		9		5	
3				8		5	3	1	4
2						7		6	8
1			8				5		9
	a	b	c	d	e	f	g	h	i

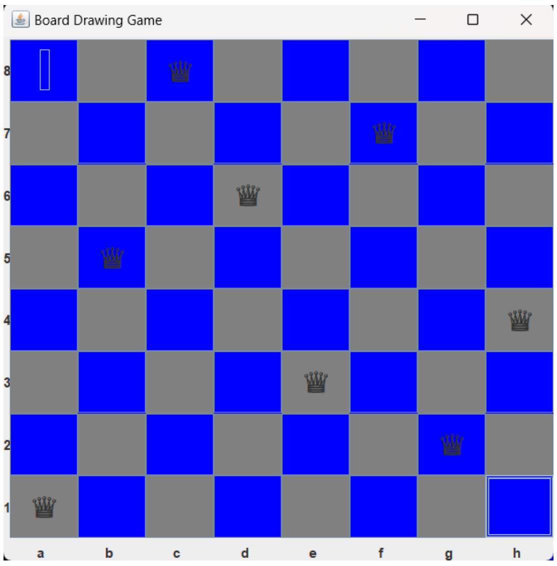


Board Drawing Game

9	6	1	9	2	7	3	4	8	5
8	2	8	3	5	1	4	7	9	6
7	5	4	7	9	8	6	1	3	2
6	3	9	6	4	5	1	8	2	7
5	1	7	5	6	2	8	9	4	3
4	8	2	4	7	3	9	6	5	1
3	7	6	2	8	9	5	3	1	4
2	9	5	1	3	4	7	2	6	8
1	4	3	8	1	6	2	5	7	9
	a	b	c	d	e	f	g	h	i

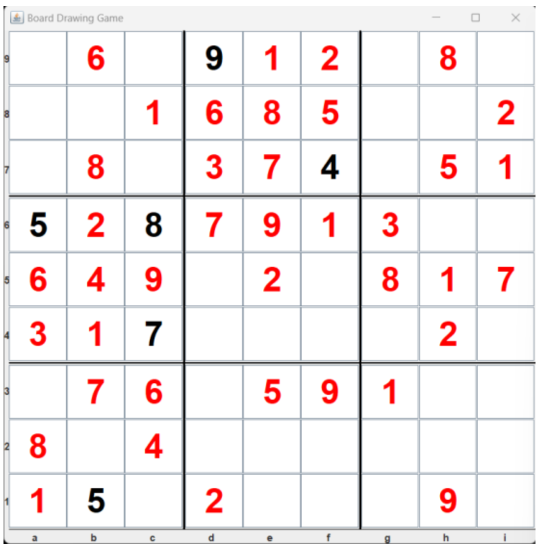
- 8-Queens Initial State





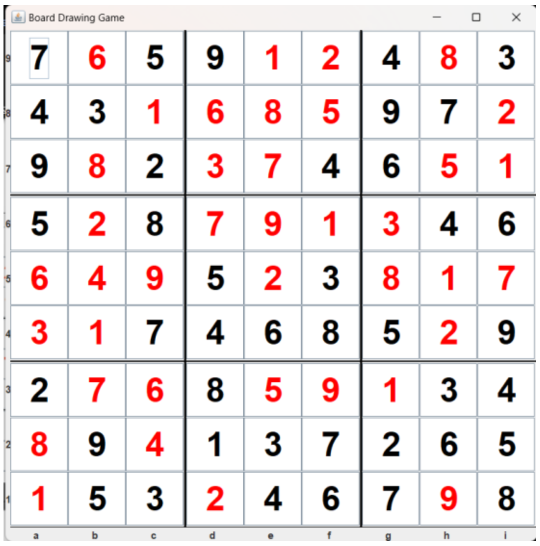
Bonus Test Cases:

- Sudoku after some moves & solvable



Board Drawing Game

	a	b	c	d	e	f	g	h	i
9		6		9	1	2		8	
8			1	6	8	5			2
7		8		3	7	4		5	1
6	5	2	8	7	9	1	3		
5	6	4	9		2		8	1	7
4	3	1	7					2	
3		7	6		5	9	1		
2	8		4						
1	1	5		2				9	

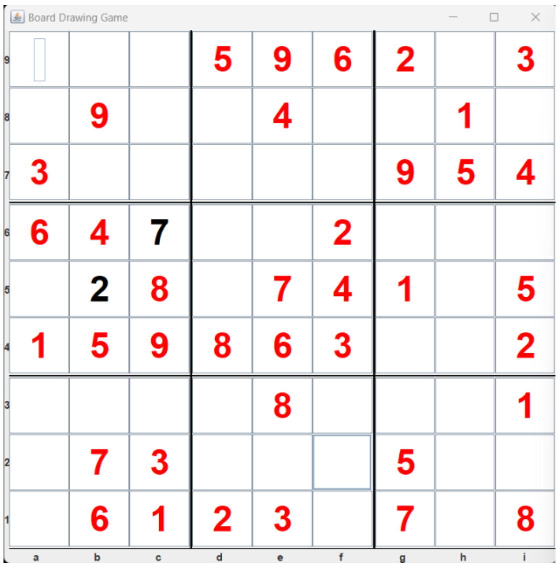


Board Drawing Game

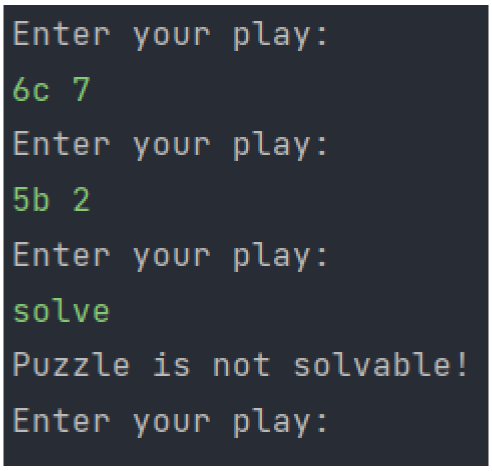
	a	b	c	d	e	f	g	h	i
9	7	6	5	9	1	2	4	8	3
8	4	3	1	6	8	5	9	7	2
7	9	8	2	3	7	4	6	5	1
6	5	2	8	7	9	1	3	4	6
5	6	4	9	5	2	3	8	1	7
4	3	1	7	4	6	8	5	2	9
3	2	7	6	8	5	9	1	3	4
2	8	9	4	1	3	7	2	6	5
1	1	5	3	2	4	6	7	9	8

Red Numbers: Initial, Black Numbers: played by user.

- **Sudoku after some moves & not solvable**

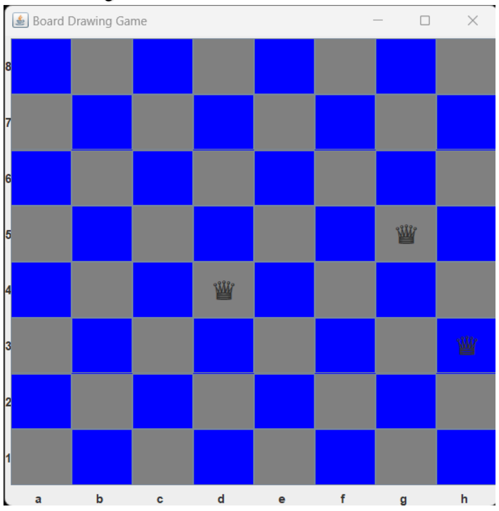


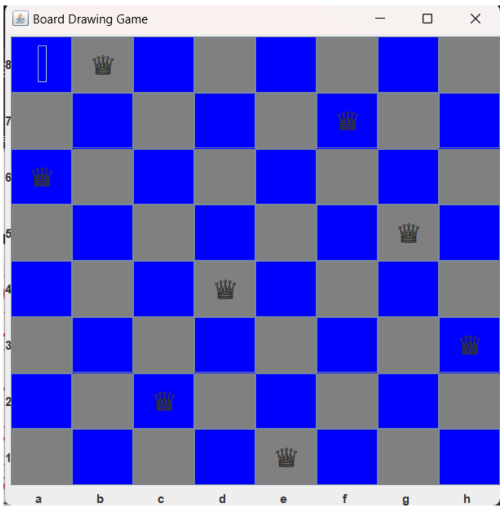
	a	b	c	d	e	f	g	h	i
9				5	9	6	2		3
8		9			4			1	
7	3						9	5	4
6	6	4	7			2			
5		2	8		7	4	1		5
4	1	5	9	8	6	3			2
3					8				1
2		7	3				5		
1		6	1	2	3		7		8



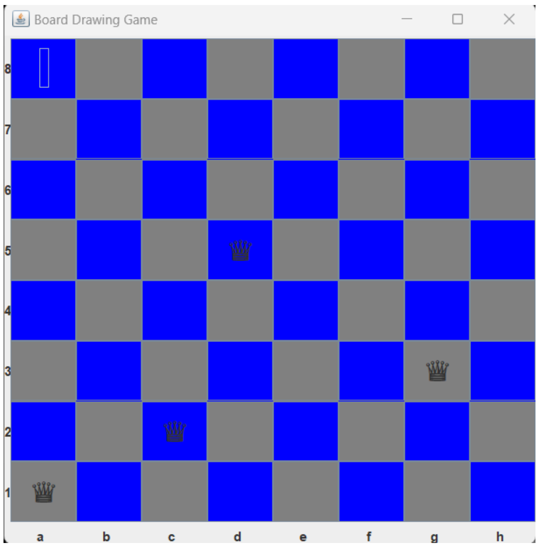
```
Enter your play:
6c 7
Enter your play:
5b 2
Enter your play:
solve
Puzzle is not solvable!
Enter your play:
```

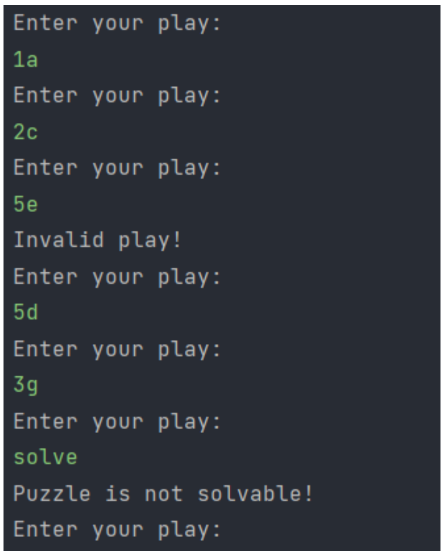
- **8-Queens after some moves & solvable**





- **8-Queens after some moves & not solvable**





```
Enter your play:
1a
Enter your play:
2c
Enter your play:
5e
Invalid play!
Enter your play:
5d
Enter your play:
3g
Enter your play:
solve
Puzzle is not solvable!
Enter your play:
```

Executable jar file:

Steps to open the executable jar file

- Open the project root directory which is called **“Board Drawing Engine”.**
- You will find the executable jar file named **“App”.**
- Open the terminal from this directory.
- write the following command: **scala App.jar**
- Write your inputs in the terminal.

Logical programming Advantages:

- The solver logic for each of the two game engines was implemented in a declarative way by just defining the game rules for each rather than specifying the detailed steps for solving the puzzle.
- All possible solutions are found so you can easily explore multiple solutions.

Logical programming Disadvantages:

- Prolog may not be as efficient as imperative programming languages.
- Lack of online education resources for such paradigm.

Refrences:

https://www.swi-prolog.org/pldoc/man?section=clpfd-n-queens
https://www.swi-prolog.org/pldoc/man?section=clpfd-sudoku