

Sudoku solver in Prolog

Rakshit Sharma

1702913086

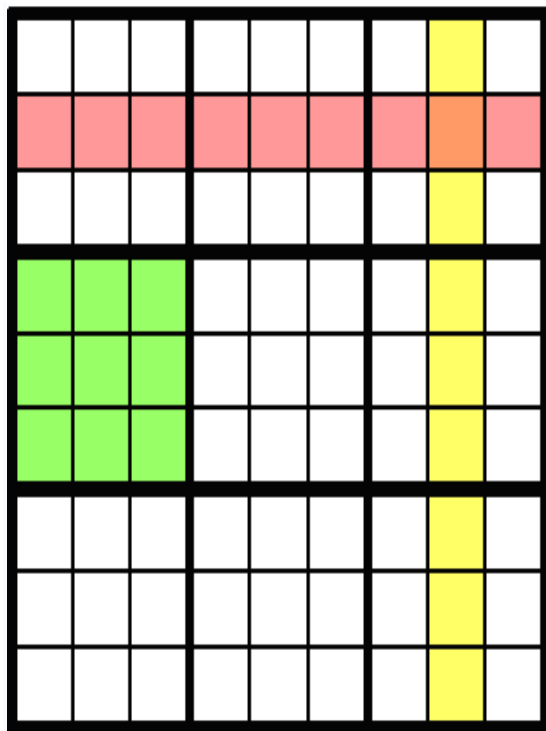
Sudoku:

Sudoku is the Japanese abbreviation of a phrase meaning the digits must remain single, also known as *Number Place*, where Su means number, doku which translates as single or bachelor.

Sudoku is not a mathematical or arithmetical puzzle. It works just as well if the numbers are substituted with letters or some other symbols, but numbers work best.

The aim of the puzzle is to enter a numerical digit from 1 through 9 in each cell of a 9×9 grid made up of 3×3 subsquares or subgrids, starting with various digits given in some cells; each row, column, and subsquares region must contain each of the numbers 1 to 9 exactly once.

Throughout this document we refer to the whole puzzle as the grid/game board, a 3×3 subgrid as a block and the individual grids that contains the number as a cell.



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

Figure : Sample Sudoku game

Rules :

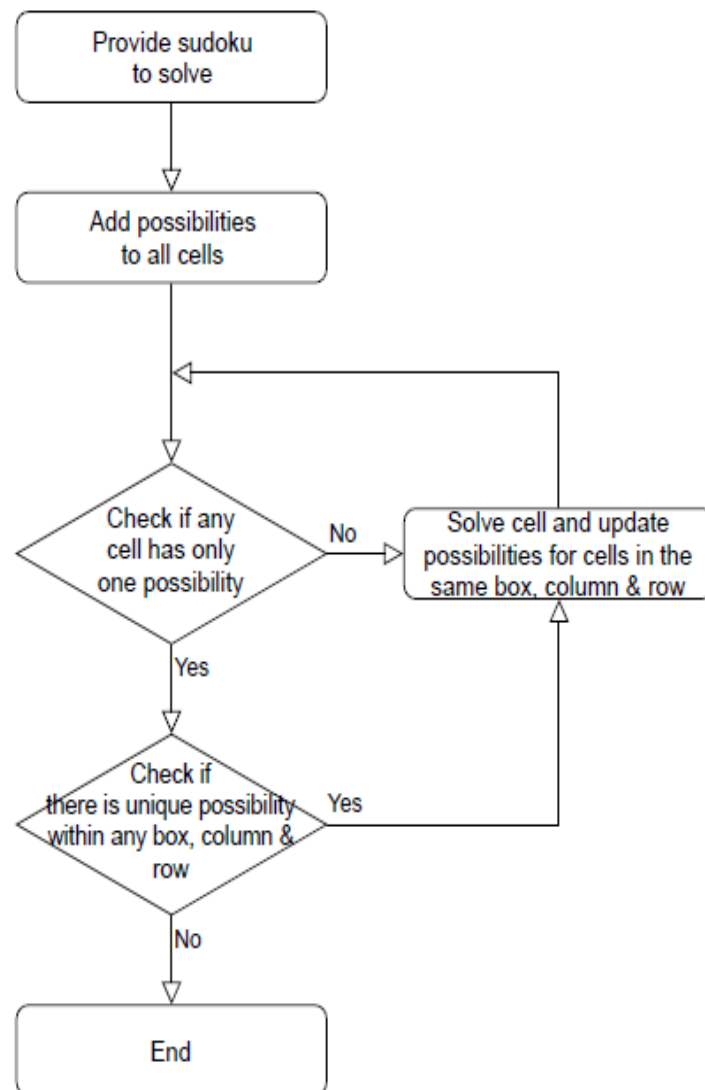
Solving a Sudoku puzzle can be rather tricky, but the rules of the game are quite simple.

Solving a sudoku puzzle does not require knowledge of mathematics; simple logic suffices.

The objective of sudoku is to enter a digit from 1 through 9 in each cell, in such a way that:

- a) Each horizontal row contains each digit exactly once
- b) Each vertical column contains each digit exactly once
- c) Each subgrid or region contains each digit exactly once

Model Design for the application :



Follow this simple steps to solve a sudoku:

- Run the prolog console

```
$ cd <THIS_REPO>
```

```
$ swi-pl
```

```
Welcome to SWI-Prolog ....
```

- Define a sudoku and run the solver

```
[sudoku].
```

```
S = [_,_,_,2,6,_,7,_,1,
      6,8,_,_,7,_,_,9,_,
      1,9,_,_,_,4,5,_,_,
      8,2,_,1,_,_,_,4,_,
      _,_,4,6,_,2,9,_,_,
      _,5,_,_,_,3,_,2,8,
      _,_,9,3,_,_,_,7,4,
      _,4,_,_,5,_,_,3,6,
      7,_,3,_,1,8,_,_,_]
```

```
solvesudoku(S, Solution).
```

or run the demonstration file:

```
[demonstration].
```

```
test.
```

The output will be something like this:

```
?- [sudoku].
```

```
true.
```

```
?- S = [_,_,_,2,6,_,7,_,1,
```

```
|      6,8,_,_,7,_,_,9,_,
```

```
|      1,9,_,_,_,4,5,_,_,
```

```
|      8,2,_,1,_,_,_,4,_,
```

```
|      _,_,4,6,_,2,9,_,_,
```

```
|      _,5,_,_,_,3,_,2,8,
```

```
|      _,_,9,3,_,_,_,7,4,
```

```
|      _,4,_,_,5,_,_,3,6,
```

```
|      7,_,3,_,1,8,_,_,_]
```

```
| solvesudoku(S, Solution).
```

```
Solve sudoku...
```

```
Solution:
```

```
[4,3,5,2,6,9,7,8,1]
```

```
[6,8,2,5,7,1,4,9,3]
```

```
[1,9,7,8,3,4,5,6,2]
```

```
[8,2,6,1,9,5,3,4,7]
```

```
[3,7,4,6,8,2,9,1,5]
```

```
[9,5,1,7,4,3,6,2,8]
```

```
[5,1,9,3,2,6,8,7,4]
```

```
[2,4,8,9,5,7,1,3,6]
```

```
[7,6,3,4,1,8,2,5,9]
```

S = Solution, Solution = [4, 3, 5, 2, 6, 9, 7, 8, 1|...].

CODE:

Sudoku.pl

```
:- use_module(library(clpfd)).
```

```
solvesudoku(Puzzle, Solution) :-
```

```
    Solution = Puzzle,
```

```
    Puzzle = [A1, B1, C1, D1, E1, F1, G1, H1, I1,
```

```
              A2, B2, C2, D2, E2, F2, G2, H2, I2,
```

```
              A3, B3, C3, D3, E3, F3, G3, H3, I3,
```

```
              A4, B4, C4, D4, E4, F4, G4, H4, I4,
```

```
              A5, B5, C5, D5, E5, F5, G5, H5, I5,
```

```
              A6, B6, C6, D6, E6, F6, G6, H6, I6,
```

```
              A7, B7, C7, D7, E7, F7, G7, H7, I7,
```

```
              A8, B8, C8, D8, E8, F8, G8, H8, I8,
```

```
              A9, B9, C9, D9, E9, F9, G9, H9, I9
```

```
    ],
```

```
    writeln("Solve sudoku..."),
```

```
    % all fields must be between 1 and 9
```

```
    Puzzle ins 1..9,
```

```
    % all rows must have only unique fields
```

```
    all_different([A1, B1, C1, D1, E1, F1, G1, H1, I1]),
```

```
    all_different([A2, B2, C2, D2, E2, F2, G2, H2, I2]),
```

```
    all_different([A3, B3, C3, D3, E3, F3, G3, H3, I3]),
```

all_different([A4, B4, C4, D4, E4, F4, G4, H4, I4]),

all_different([A5, B5, C5, D5, E5, F5, G5, H5, I5]),

all_different([A6, B6, C6, D6, E6, F6, G6, H6, I6]),

all_different([A7, B7, C7, D7, E7, F7, G7, H7, I7]),

all_different([A8, B8, C8, D8, E8, F8, G8, H8, I8]),

all_different([A9, B9, C9, D9, E9, F9, G9, H9, I9]),

% all columns must have only unique fields

all_different([A1, A2, A3, A4, A5, A6, A7, A8, A9]),

all_different([B1, B2, B3, B4, B5, B6, B7, B8, B9]),

all_different([C1, C2, C3, C4, C5, C6, C7, C8, C9]),

all_different([D1, D2, D3, D4, D5, D6, D7, D8, D9]),

all_different([E1, E2, E3, E4, E5, E6, E7, E8, E9]),

all_different([F1, F2, F3, F4, F5, F6, F7, F8, F9]),

all_different([G1, G2, G3, G4, G5, G6, G7, G8, G9]),

all_different([H1, H2, H3, H4, H5, H6, H7, H8, H9]),

all_different([I1, I2, I3, I4, I5, I6, I7, I8, I9]),

% all squares must have only unique fields

all_different([A1, A2, A3, B1, B2, B3, C1, C2, C3]),

all_different([A4, A5, A6, B4, B5, B6, C4, C5, C6]),

all_different([A7, A8, A9, B7, B8, B9, C7, C8, C9]),

```
all_different([D1, D2, D3, E1, E2, E3, F1, F2, F3]),
all_different([D4, D5, D6, E4, E5, E6, F4, F5, F6]),
all_different([D7, D8, D9, E7, E8, E9, F7, F8, F9]),
all_different([G1, G2, G3, H1, H2, H3, I1, I2, I3]),
all_different([G4, G5, G6, H4, H5, H6, I4, I5, I6]),
all_different([G7, G8, G9, H7, H8, H9, I7, I8, I9]),

label(Solution), % resolve variables

% print out solution

writeln(""),
writeln("Solution:"),
writeln([A1, B1, C1, D1, E1, F1, G1, H1, I1]),
writeln([A2, B2, C2, D2, E2, F2, G2, H2, I2]),
writeln([A3, B3, C3, D3, E3, F3, G3, H3, I3]),
writeln([A4, B4, C4, D4, E4, F4, G4, H4, I4]),
writeln([A5, B5, C5, D5, E5, F5, G5, H5, I5]),
writeln([A6, B6, C6, D6, E6, F6, G6, H6, I6]),
writeln([A7, B7, C7, D7, E7, F7, G7, H7, I7]),
writeln([A8, B8, C8, D8, E8, F8, G8, H8, I8]),
writeln([A9, B9, C9, D9, E9, F9, G9, H9, I9]),
writeln("").
```

Demonstration.pl

```
:- [sudoku]. % include sudoku solver
```

```
test() :-
```

```
    % define a sudoku to solve (_ are the unknown fields)
```

```
    S = [9,8,_,7,_,_,6,_,_,,
```

```
         7,5,_,_,_,_,_,9,_,,
```

```
         _,_,6,_,_,_,_,_,_,,
```

```
         6,4,_,_,_,_,_,_,_,,
```

```
         _,_,9,6,_,_,_,5,_,,
```

```
         _,_,_,_,_,3,_,_,_,,
```

```
         _,_,7,9,_,_,_,8,3,_,
```

```
         _,_,5,8,_,_,_,9,6,_,,
```

```
         _,_,_,_,2,_,_,_,1],
```

```
    solvesudoku(S, Solution),
```

```
    write("Raw solution: "),
```

```
    writeln(Solution),
```

```
    writeln("").
```


OUTPUT:

Running Sudoku.pl

```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.1)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

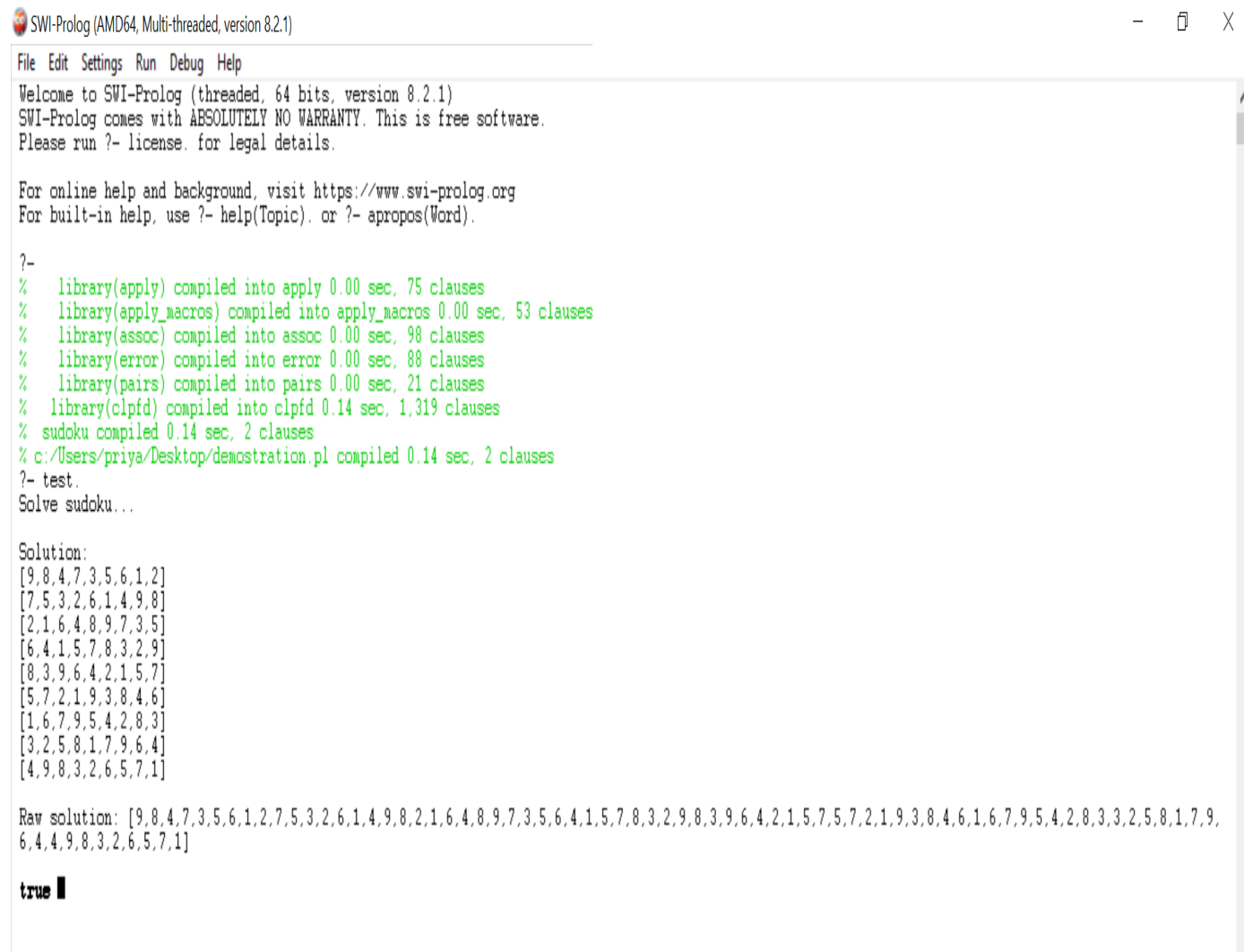
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?-
% library(apply) compiled into apply 0.00 sec, 75 clauses
% library(apply_macros) compiled into apply_macros 0.02 sec, 53 clauses
% library(assoc) compiled into assoc 0.02 sec, 98 clauses
% library(error) compiled into error 0.00 sec, 88 clauses
% library(pairs) compiled into pairs 0.02 sec, 21 clauses
% library(clpfd) compiled into clpfd 0.20 sec, 1,319 clauses
% c:/Users/priya/Desktop/sudoku.pl compiled 0.20 sec, 2 clauses
?- S = [_,_,_2,6,_7,_1,
        6,8,_,_7,_9,_,
        1,9,_,_4,5,_,_
        8,2,_1,_,_4,_,
        _4,6,_2,9,_,_
        _5,_,_3,_2,8,
        _9,3,_,_7,4,
        _4,_5,_3,6,
        7,_3,_1,8,_,_],
    solvesudoku(S, Solution).
Solve sudoku...

Solution:
[4,3,5,2,6,9,7,8,1]
[6,8,2,5,7,1,4,9,3]
[1,9,7,8,3,4,5,6,2]
[8,2,6,1,9,5,3,4,7]
[3,7,4,6,8,2,9,1,5]
[9,5,1,7,4,3,6,2,8]
[5,1,9,3,2,6,8,7,4]
[2,4,8,9,5,7,1,3,6]
[7,6,3,4,1,8,2,5,9]

S = Solution, Solution = [4, 3, 5, 2, 6, 9, 7, 8, 1|...].
?-
```

Running Demonstration.pl

A screenshot of the SWI-Prolog (AMD64, Multi-threaded, version 8.2.1) window. The window has a menu bar with 'File', 'Edit', 'Settings', 'Run', 'Debug', and 'Help'. The main text area shows the following content:

```

Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?-
%   library(apply) compiled into apply 0.00 sec, 75 clauses
%   library(apply_macros) compiled into apply_macros 0.00 sec, 53 clauses
%   library(assoc) compiled into assoc 0.00 sec, 98 clauses
%   library(error) compiled into error 0.00 sec, 88 clauses
%   library(pairs) compiled into pairs 0.00 sec, 21 clauses
%   library(clpfd) compiled into clpfd 0.14 sec, 1,319 clauses
%   sudoku compiled 0.14 sec, 2 clauses
% c:/Users/priya/Desktop/demostration.pl compiled 0.14 sec, 2 clauses
?- test.
Solve sudoku...

Solution:
[9,8,4,7,3,5,6,1,2]
[7,5,3,2,6,1,4,9,8]
[2,1,6,4,8,9,7,3,5]
[6,4,1,5,7,8,3,2,9]
[8,3,9,6,4,2,1,5,7]
[5,7,2,1,9,3,8,4,6]
[1,6,7,9,5,4,2,8,3]
[3,2,5,8,1,7,9,6,4]
[4,9,8,3,2,6,5,7,1]

Raw solution: [9,8,4,7,3,5,6,1,2,7,5,3,2,6,1,4,9,8,2,1,6,4,8,9,7,3,5,6,4,1,5,7,8,3,2,9,8,3,9,6,4,2,1,5,7,5,7,2,1,9,3,8,4,6,1,6,7,9,5,4,2,8,3,3,2,5,8,1,7,9,6,4,4,9,8,3,2,6,5,7,1]

true
```