

5	6	3	8	7	9	2	1	4	5	6	3	8	7	9	2	1	4
7	1	9	4	2	3	6	5	8	7	1	9	4	2	3	6	5	8
2	S	U	D	O	K	U	9	7	2	8	4	5	6	1	3	9	7
4	2	S	O	L	V	E	R	3	4	2	6	1	5	7	9	8	3
1	9	5	6	3	8	4	7	2	1	9	5	6	3	8	4	7	2
8	3	7	2	9	4	1	6	5	8	3	7	2	9	4	1	6	5
9	4	8	3	1	5	7	2	6	9	R	O	N	1	5	7	2	6
6	5	1	7	4	2	8	3	9	6	5	W	O	O	D	8	3	9
3	7	2	9	8	6	5	4	1	3	7	2	9	8	6	5	4	1
6	5	1	7	4	2	8	3	9	6	5	1	7	4	2	8	3	9
8	3	7	2	9	4	1	6	5	8	3	7	2	9	4	1	6	5

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

Each Row needs  
to have 1-9

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

Each column  
needs to have  
1-9

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

Each 3x3 box  
needs to have  
1-9

```
[[7,3,9,0,0,0,0,0,0],
 [0,1,4,3,0,8,0,0,0],
 [0,0,0,0,4,9,0,5,0],
 [0,0,0,0,2,0,7,3,0],
 [9,7,0,0,0,4,6,0,2],
 [6,4,2,1,7,3,0,0,8],
 [0,0,7,8,0,0,4,6,0],
 [3,0,0,0,9,0,0,0,0],
 [4,0,0,6,5,2,0,7,0]]
```

=

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

A 9x9 sudoku board can be represented by a list of lists of integers.

0 Can be used to represent a blank cell.

# Single Threaded Solution

There are two parts to the Solve Function.


1. Generate Possible Options
2. Solve Each Option

# 1. Generate Possible Options

The function finds the first empty cell.


It then tests each value 1-9.

If the value is not in the same row, column, or box, then it is a valid option.

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


## 2. Solve Each Option

1 2 3 4 5 6 7 8 9



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

1 2 3 4 5 6 7 8 9



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

This continues until there are no more valid options, or a solution is found

# Optimizing the Solve Function

Instead of choosing the first empty cell, explore the cell with the least amount of available options.

1 2 3 4 5 6 7 8 9

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



# Multi Threaded Solution

The multi threaded solution generates options the same way as the single threaded solution.

For each option generated, the function starts a thread dedicated to solving that option.

The threads have access to shared data

**solution** - This is where the solution is stored  
(originally null)

**num\_threads** - The current number of alive worker  
threads

**max\_threads** - The maximum number of threads we  
allow to exist

**mutex** - used to control access to shared data

**cv** - used to alert master thread when solution is found

# Master Thread

- Creates the first worker thread
- Listens on `cv` to check if a `solution` was found
- Ends all threads if a solution is found
- Ends if all worker threads are terminated (no solution)

# Worker Thread

- Solves input board
- When generating options, if `num_threads < max_threads`, the worker thread can start another worker thread
- Thread ends when it has explored all options or found a solution.
- Every time a thread ends or starts another thread, it updates `num_threads`.