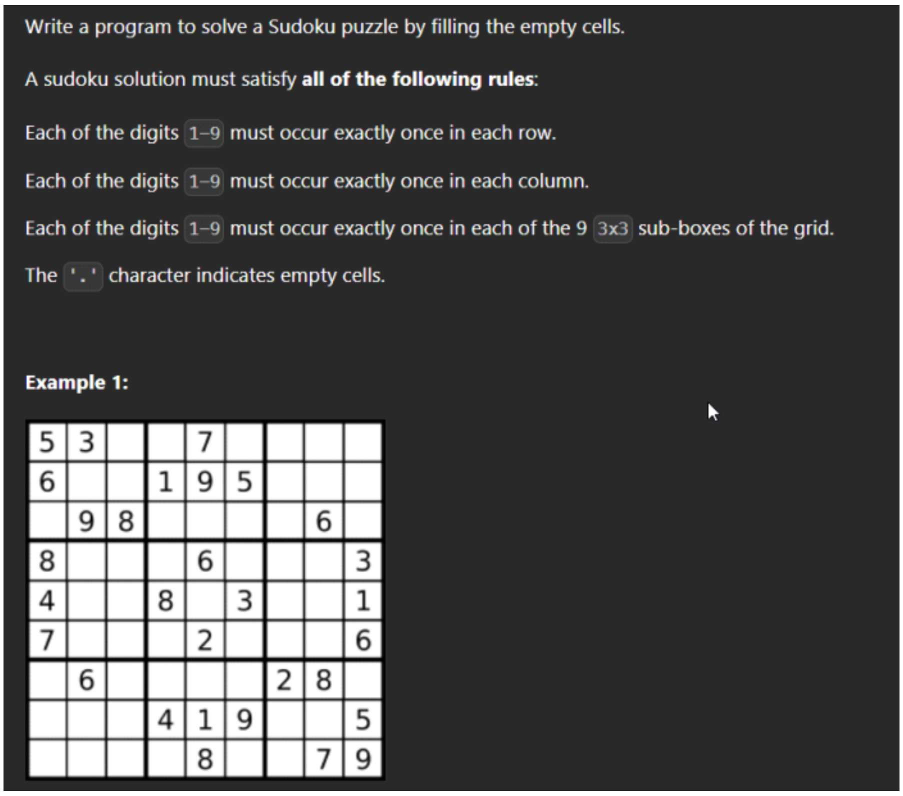
Day 7

30 November 2022 19:32



Write a program to solve a Sudoku puzzle by filling the empty cells.

A sudoku solution must satisfy **all of the following rules**:

Each of the digits `1-9` must occur exactly once in each row.

Each of the digits `1-9` must occur exactly once in each column.

Each of the digits `1-9` must occur exactly once in each of the 9 `3x3` sub-boxes of the grid.

The `'.'` character indicates empty cells.

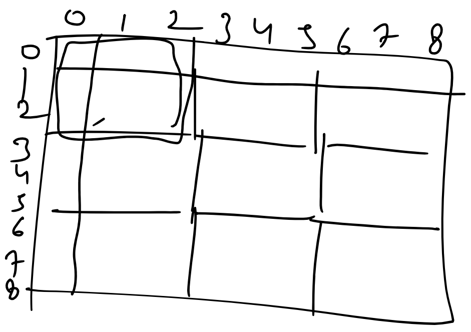
Example 1:

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

1 – 9 9 x 9

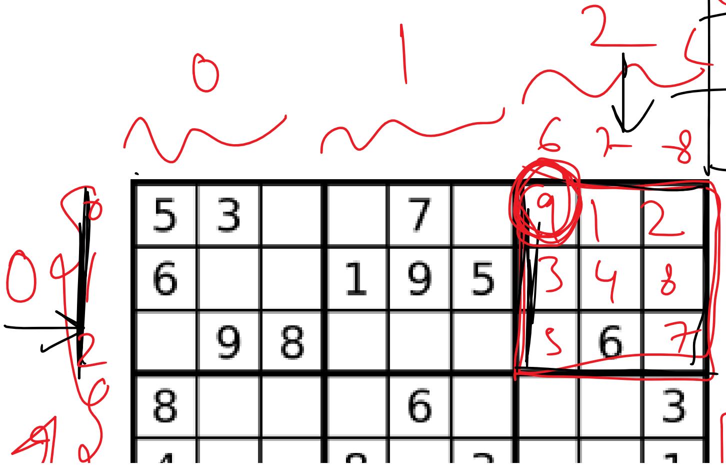
हर row 1 – 9 one time

हर colum 1 – 9 one time



9 (3x3)

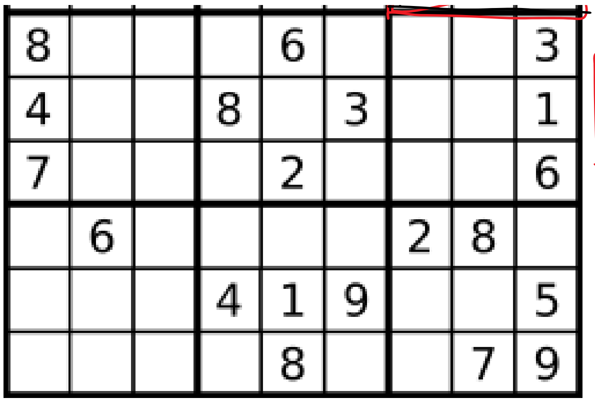
(0,6)	(0,7)	(0,8)
(1,6)	(1,7)	(1,8)
(2,6)	(2,7)	(2,8)



row = 0

col = 6

[(row) +1]



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

1

2

$$\left[3\times\left(\frac{row}{3}\right)+i\right]$$

$$\left[3\times\left(\frac{col}{3}\right)+j\right]$$

(0,6)

$3\times\left(\frac{0}{3}\right)\longrightarrow$ 0th 0

$3\times\left(\frac{6}{3}\right)\longrightarrow$ 2nd $\Rightarrow$ 6

+0 +6

(0,0)	(0,1)	(0,2)
(1,0)	(1,1)	(1,2)
(2,0)	(2,1)	(2,2)

, puzzle))

of du

	0th	1st	2nd
0th			
1th			
2nd			

(/) x 3 -

$$\left(\frac{0}{3}\right)^{\times 3} -$$

$$\left(\frac{6}{3}\right)^{\times 3}$$

```cpp
class Solution {
public:
    bool is_valid_position(int row, int col, char ch,vector<vector<char>> &board){
        // row
        for(int i = 0;i < 9;i++){
            if(board[row][i] == ch){
                return false;
            }
            if(board[i][col] == ch){
                return false;
            }
        }
        // gris 3*3
        for(int i = 0;i < 3;i++){
            for(int j = 0;j < 3;j++){
                if(board[i +(row/3)*3][j + (col/3)*3] == ch){
                    return false;
                }
            }
        }
        return true;
    }
    bool solver(vector<vector<char>> &board){
        for(int  row =0;row < 9;row++){
            for(int col = 0;col < 9;col++){
                if(board[row][col] == '.'){
                    for(int pos = 1;pos <= 9;pos++){
                         if(is_valid_position(row,col,'0'+pos,board)){
                            board[row][col] = '0' + pos;
                            if(solver(board)) return true;
                            board[row][col] = '.';
                         }
                    }
                    return false;
                }
            }
        }
        return true;
    }
    void solveSudoku(vector<vector<char>>& board) {
        solver(board);
    }
};
```