CRACK A HACK

SUDOKU

-Project Euler #96

Algorithmic Problem Solving
Course code: 17ECSE309

By,
Shivakumar V H
01FE15BEC181

1. Problem Statement:

Su Doku (Japanese meaning *number place*) is the name given to a popular puzzle concept. Its origin is unclear, but credit must be attributed to Leonhard Euler who invented a similar, and much more difficult, puzzle idea called Latin Squares. The objective of Su Doku puzzles, however, is to replace the blanks (or zeros) in a 9 by 9 grid in such that each row, column, and 3 by 3 box contains each of the digits 1 to 9. Below is an example of a typical starting puzzle grid and its solution grid.

A well constructed Su Doku puzzle has a unique solution and can be solved by logic, although it may be necessary to employ "guess and test" methods in order to eliminate options (there is much contested opinion over this). The complexity of the search determines the difficulty of the puzzle; the example above is considered *easy* because it can be solved by straight forward direct deduction.

You are given a number of Su Doku. All of them could be solved without guessing and even backtracking. Surely, you may write every solution that passes tests and fits into contraints.

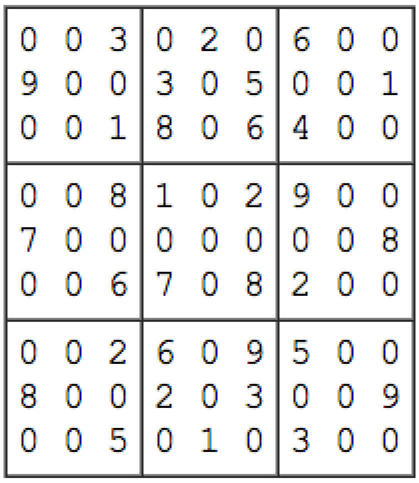
Input Format

9 lines each containg 9 characters '0'-'9'. '0' means that the place is empty otherwise the place contains corresponding digit.

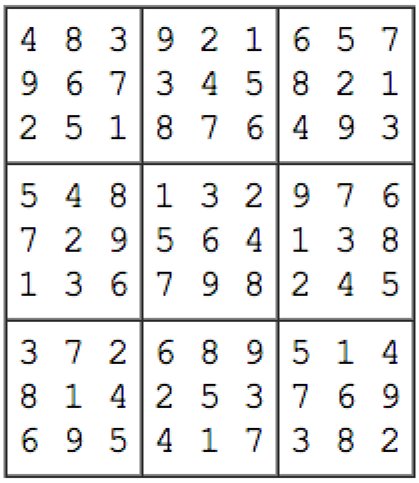
Output Format

Output the result in the same format as input with no zeroes left.

2. Example:



0	0	3	0	2	0	6	0	0
9	0	0	3	0	5	0	0	1
0	0	1	8	0	6	4	0	0
0	0	8	1	0	2	9	0	0
7	0	0	0	0	0	0	0	8
0	0	6	7	0	8	2	0	0
0	0	2	6	0	9	5	0	0
8	0	0	2	0	3	0	0	9
0	0	5	0	1	0	3	0	0



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

3. Code:

```
#include <cstdio>
#include <iostream>
#include <sstream>
#include <string>
#include <cmath>
#include <cassert>
#include <algorithm>
#include <vector>
#include <set>
#include <map>
#include <deque>
using namespace std;
typedef long long ll;
typedef pair<double, double> dd;
typedef pair<int, int> ii;
typedef pair<int, ii> iii;
typedef vector<int> vi;
typedef vector<ii> vii;
typedef vector<vi> vvi;
typedef vector<vii> vvii;


vector<string> g;
const int n = 9;

ii next_cell(int mi, int mj){
	if(mj != n-1){
		return ii(mi, mj+1);
	}else if(mi != n-1){
		return ii(mi+1, 0);
	}else{
		return ii(-1, -1);
	}
}

bool can_place(int mi, int mj, int num){
	for(int i=0;i<n;i++){
		if(g[mi][i] == num+'0') return false;
		if(g[i][mj] == num+'0') return false;
	}
	// check 3x3 area
	int ni = mi/3;
	int nj = mj/3;
	ni *= 3;
	nj *= 3;
	for(int i=ni;i<ni+3;i++){
		for(int j=nj;j<nj+3;j++){
			if(g[i][j] == num+'0') return false;
		}
	}
	return true;

}

bool solve(int mi, int mj){
	if(mi == -1 && mj == -1) return true; // past last cell

	ii nxt = next_cell(mi, mj);

	if(g[mi][mj] != '0'){
		return solve(nxt.first, nxt.second);
	}else{
		for(int i=1;i<=9;i++){
			if(can_place(mi, mj, i)){
				g[mi][mj] = '0'+i;
				int solved = solve(nxt.first, nxt.second);
				if(solved){
					return true;
				}
				g[mi][mj] = '0';
			}
		}
	}
	return false;
}
```

```
int main(){
        g = vector<string>(n);
        for(int i=0;i<n;i++){
                cin >> g[i];
        }

        solve(0, 0);

        for(int i=0;i<n;i++){
                cout << g[i] << endl;
        }
}
```

4. References:

- **https://www.mathblog.dk/project-euler-96-su-doku/**