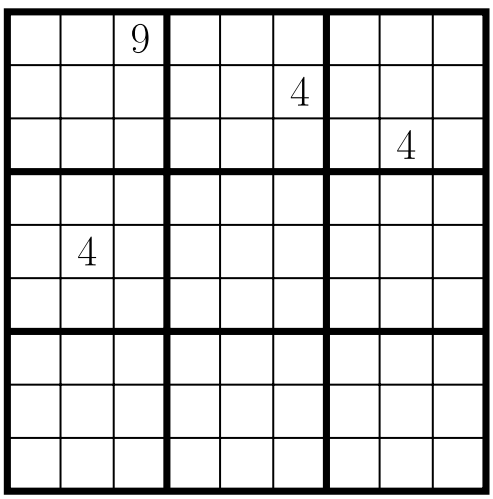
COCI '08 Contest 3 #3 Cross

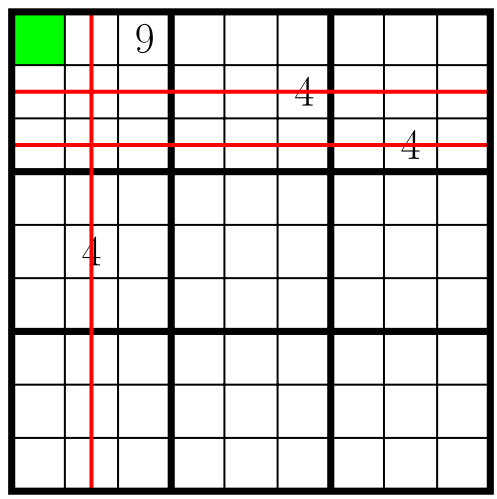
Time limit: 0.6s **Memory limit:** 32M

In the game of Sudoku, the objective is to place integers between 1 and 9 (inclusive) into a 9×9 grid so that each row, each column, and each of the nine 3×3 boxes contains all nine numbers. The starting board is partially filled in so that it is possible to logically deduce the values of the other cells. Sudoku puzzles range in difficulty, and complex analysis methods are required to solve the hardest puzzles. In this problem, however, you will implement one of the simplest methods, cross-hatching.

In cross-hatching, we select one of the nine numbers and, for each of its occurrences in the grid, cross out the corresponding row, column and 3×3 box. Now look for any 3×3 boxes where there is only one possible placement for the number and place it there.

The first image below shows a very sparsely filled in Sudoku grid. However, even in this grid it is possible to deduce using cross-hatching that the number in the top left cell is 4, as illustrated in the second image.





You will be given a partially filled-in grid. Your task is to repeatedly apply the cross-hatching method for different numbers until no more deductions can be made about any number.

The initial placement of the numbers in the grid may be invalid. It is also possible that there will be no available cell for a number in a 3×3 box. In both cases, you are to report an error.

Input Specification

Input will consist of 9 lines, each containing exactly 9 characters. Each character will either be a digit between 1 and 9, or a period `.` denoting an empty cell.

Output Specification

If the input is valid and there is no contradiction while solving, you should output the grid in the same format it was given in, with cells filled in if their value can be deduced using cross-hatching. Otherwise, output `ERROR`.

Sample Input 1

```
..9......
.....4...
.......4.
.........
.4.......
.........
.........
.........
.........
```

Sample Output 1

```
4.9......
.....4...
.......4.
.........
.4.......
.........
.........
.........
.........
```

Sample Input 2

```
...1...6.
18...9...
..7.642..
2.9..6.5.
.43...72.
.6.3..9.1
..265.1..
...2...97
.5...3...
```

Sample Output 2

```
524137869
186529473
397864215
219476358
843915726
765382941
972658134
638241597
451793682
```

Sample Input 3

```
1........
..1......
.......1.
.........
.........
.........
.........
.........
.........
```

Sample Output 3

```
ERROR
```

Sample Input 4

```
........2
....1....
1........
......1..
.........
.........
.........
.......1.
.........
```

Sample Output 4

```
ERROR
```