LACS-Elite-Part012

Solved Challenges 0/1

Back To Challenges List

Solve Sudoku

ID:12946 **Solved By 1 Users**

The program must accept an integer matrix of size **9x9** representing a sudoku as the input. The sudoku matrix contains the integers from 0 to 9 where **0** represents the **empty cells**. If the sudoku matrix is valid, the program must fill in the empty cells of the sudoku matrix and print it as the output. Else the program must print **Not Solved** as the output.

Sudoku:
Sudoku is a logic-based, combinatorial number-placement puzzle. The objective is to fill a 9×9 grid with digits so that each column, each row, and each of the nine 3×3 subgrids that compose the grid contain all of the digits from 1 to 9.

Input Format:
The first 9 lines each contain 9 integers separated by a space.

Output Format:
The first 9 lines each contain 9 integers separated by a space or the first line contains Not Solved.

Example Input/Output 1:
Input:
0 0 0 2 6 0 7 0 1
6 8 0 0 7 0 0 9 0
1 9 0 0 0 4 5 0 0
8 2 0 1 0 0 0 4 0
0 0 4 6 0 2 9 0 0
0 5 0 0 0 3 0 2 8
0 0 9 3 0 0 0 7 4
0 4 0 0 5 0 0 3 6
7 0 3 0 1 8 0 0 0

Output:
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

Example Input/Output 2:
Input:
0 6 0 3 0 0 8 0 4
5 3 7 0 9 0 0 0 0
0 4 0 0 0 6 3 0 7
0 9 0 0 5 1 2 3 8
0 0 0 0 0 0 0 0 0
7 1 3 6 2 0 0 4 0
3 0 6 4 0 0 0 1 0
0 0 0 0 6 0 5 2 3
1 0 2 0 0 3 0 8 0

Output:
Not Solved

Max Execution Time Limit: 500 millisecs

Ambiance

Java (12.0)

Reset

```
1  import java.util.*;
2  class Slot{
3      int r,c;
4  }
5
6  public class Hello {
7      static final int R=9, C=9;
8      public static void main(String[] args) {
9          Scanner sc = new Scanner(System.in);
10         int matrix[][] = new int[R][C];
11         for(int row=0;row<R;row++)
12         {
13             for(int col=0;col<C;col++)
14             {
15                 matrix[row][col] = sc.nextInt();
16             }
17         }
18
19         if(solve(matrix))
20         {
21             for(int row=0;row<R;row++)
```

```
22                  for(int col=0;col<C;col++)
23                      System.out.print(matrix[row][col]+ " ");
24                  System.out.println("");
25          }
26          else
27              System.out.println("Not Solved");
28      }
29
30      public static boolean solve(int[][] matrix)
31      {
32          Slot slot = getFreeSlot(matrix);
33          if(slot == null)
34          return true;
35
36          for(int digit=1;digit<=9;digit++)
37          {
38              if(canFillRow(matrix,slot,digit) && canFillCol(matri
                    ,digit) && canFillSubmatrix(matrix,slot,digit))
39              {
40                  matrix[slot.r][slot.c] = digit;
41                  if(solve(matrix))
42                  return true;
43                  else
44                  matrix[slot.r][slot.c] = 0;
45
46              }
47          }
48
49          return false;
50      }
51
52      public static Slot getFreeSlot(int[][]matrix)
53      {
54          for(int row=0;row<R;row++)
55          for(int col=0;col<C;col++)
56          {
57              if(matrix[row][col]==0)
58              {
59                  Slot slot = new Slot();
60                  slot.r = row;
61                  slot.c = col;
62                  return slot;
63              }
64          }
65
66          return null;
67      }
68
69      public static boolean canFillRow(int[][]matrix, Slot slot,i
70      {
```

```
71          for(int col=0;col<C;col++)
72          {
73              if(matrix[slot.r][col] == digit)
74              return false;
75          }
76          return true;
77      }
78
79      public static boolean canFillCol(int[][]matrix,Slot slot,int
80      {
81          for(int row=0;row<R;row++)
82          {
83              if(matrix[row][slot.c] == digit)
84              return false;
85          }
86          return true;
87      }
88
89      public static boolean canFillSubmatrix(int[][]matrix,Slot s
            digit)
90      {
91          int startRow = (slot.r/3)*3;
92          int startCol = (slot.c/3)*3;
93
94          for(int row=startRow;row<=startRow+2;row++)
95          {
96              for(int col = startCol;col<=startCol+2;col++)
97              {
98                  if(matrix[row][col] == digit)
99                  return false;
100             }
101         }
102         return true;
103     }
104
105
106
```

Code did not pass the execution

Hello.java:2: error: class Slot is public, should be declared in a file named Slot.java
public class Slot{

^

1 error

Save Run

Run with a custom test case (Input/Output)