Logic Programming: Solving a Puzzle

We hope that you know Sudoku. In this exercise we discuss a Mini-Sudoku, which is a simplified Sudoku.

A Mini-Sudoku consists of a table with 3 rows and 3 columns. It's the aim to place in every of the nine fields one number from the set {1, 2, 3}. However there are some constraints that have to be considered: In every row and in every column the numbers need to be different.

1	2	3
2	3	1
3	1	2

The Mini-Sudoku may start with some numbers fixed at some fields.

Exercise

1. Think about to write a program in a (object-oriented or procedural) programming language like JAVA, C++, Basic, etc. which solves the puzzle automatically.

   SOLUTION: I have no idea. But if you want to implement it, it will need a lot of efforts.

2. Write a PROLOG Program that solves the Mini-Sudoku.

   a. How can a solution be encoded? I.e. how can we represent that this is a valid combination of numbers?

   b. To check a valid combination of numbers which conditions are needed to be satisfied?

Solution for 2.a) It has to be encoded into a predicate. It's true if the combination of numbers is correct. E.g.

```
solve(1,2,3,
      2,3,1,
      3,1,2).
```

One possible SOLUTION for b):

```
check(A1,A2,A3,
      B1,B2,B3,
      C1,C2,C3) :- different_numbers(A1,A2,A3),
                   different_numbers(B1,B2,B3),
                   different_numbers(C1,C2,C3),
                   different_numbers(A1,B1,C1),
                   different_numbers(A2,B2,C2),
                   different_numbers(A3,B3,C3).

different_numbers(X,Y,Z) :- X =\= Y, Y =\= Z, X =\= Z.


numbers(A1,A2,A3,
        B1,B2,B3,
        C1,C2,C3) :- select_number(A1),
                     select_number(A2),
                     select_number(A3),
                     select_number(B1),
                     select_number(B2),
                     select_number(B3),
                     select_number(C1),
                     select_number(C2),
                     select_number(C3).
select_number(1).
select_number(2).
select_number(3).

solve(A1,A2,A3,
      B1,B2,B3,
      C1,C2,C3) :- numbers(A1,A2,A3,
                           B1,B2,B3,
                           C1,C2,C3),
                   check(A1,A2,A3,
                         B1,B2,B3,
                         C1,C2,C3).

% solve(A1,A2,A3,B1,B2,B3,C1,C2,C3).
```