ModRef 2022: Model and Solve Competition
Number Chain

1 Problem Statement

This is a puzzle. The aim is to fill a grid with consecutive numbers that connect horizontally, vertically or diagonally.

You are given clues for an $n \times m$ grid that you need to fill the cells with numbers from 0 up to nm. A cell with clue with -1 can only be filled with a 0. A cell with clue 0 can be filled with any number (including 0). Each cell with positive clue can only be filled with the number given or 0. If a positive number appears in a clue then it cannot be placed in any other cell, either the given cell has that number of the number never appears. The aim is to fill in the longest sequence of numbers $1..maxnum$.

The rules are that each number from $1..maxnum$ appears exactly once, and any adjacent positive numbers k and $k+1$ must appear adjacent in the grid, which means either horizontally, vertically or diagonally. Other cells must be filled with 0s.

Consider the example puzzle for a 4×4 grid below

1	0	0	10
0	0	-1	0
0	-1	0	0
7	0	0	16

A potential solution is

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

Note how the clues that are used (1 and 7) appear in their correct position, and unused and negative clues are replaced with zeros. Note that we can't extend this solution to $maxnum = 10$ since the only place a 10 can be positioned is at the top right, which is too far from the 7. This solution uses 1 to $maxnum = 7$.

An optimal solution with $maxnum = 9$.

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

The data is available in JSON format

```
int: n; % no of ROWS
set of int: ROW = 1..n;
int: m; % no of COLS
set of int: COL = 1..m;
set of int: CLUE = -1..m*n;
array[ROW,COL] of CLUE: clue;
```

For example, the data for the example above is given as

```
{
  "n" : 4,
  "m" : 4,
  "clue" : [[1, 0, 0, 10], [0, 0, -1, 0], [0, -1, 0, 0], [7, 0, 0, 16]]
}
```

The output of your model should be in JSON format, a 2D array with n rows and m columns with name "x". The optimal solution above is given by

```
{
  "x" : [[1, 3, 4, 0], [2, 5, 0, 0], [6, 0, 9, 0], [7, 8, 0, 0]]
}
```