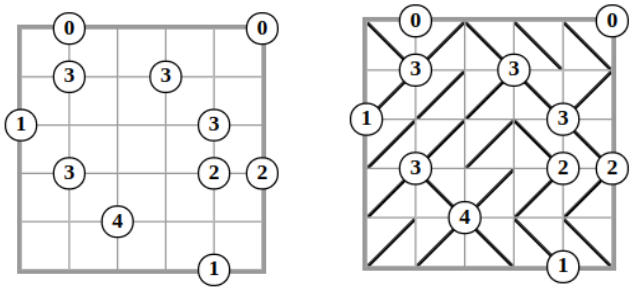
Assignment 2:
Solving Slant Puzzle using CSP

Due Sunday, 23 April, 11:30pm

Slant puzzle is a game that is played on a square grid, where some intersections contain circles with numbers, as illustrated in Figure 1 (left).[1] The aim is to draw in each grid cell one diagonal line slanting to the right or to the left, as illustrated in the example in Figure 1 (center), subject to the following rules:

- The number inside each circle is exactly the number of diagonal lines that intersect at that circle.
- The diagonal lines do not form a loop.



*	0	*	*	*	0
*	3	*	3	*	*
1	*	*	*	3	*
*	3	*	*	2	2
*	*	4	*	*	*
*	*	*	*	1	*

Figure 1: A Slant puzzle instance (left), its solution (center), and its representation as a matrix (right).

For the purpose of demos, present puzzle instances by matrices, as shown for the example in Figure 1(right):[2]

- For every (i, j) and for every number k, the matrix element (i, j) is labeled by k iff the grid intersection (i, j) is labeled by a circled number k.
- For every (i, j), the matrix element (i, j) is labeled by "$*$" iff the grid intersection (i, j) does not contain a circle with a number.

For the purpose of demos, present your solution in an easy-to-understand format.

[1] Slant puzzle: `https://www.puzzle-slant.com/`

[2] In a matrix of size $n \times n$, the rows are labeled $1, 2, ..., n$ starting from the top, and the columns are labeled $1, 2, ..., n$ starting from the left. Then, (i, j) denotes the matrix element in row i and column j. In a grid of size $n \times n$, the horizontal grid lines are labeled $1, 2, ..., n$ starting from the top, and the vertical grid lines are labeled $1, 2, ..., n$ starting from the left. Then, (i, j) denotes the intersection of the horizontal grid line i and the vertical grid line j.

What to do The assignment consists of three parts:

1. (20 points) Model the Slant puzzle as a Constraint Satisfaction Problem (CSP).
2. (60 points, provided that part 1 is completed) Based on this CSP model, solve the Slant puzzle using one of the following state-of-the-art CP solvers:
   - MiniZinc (with Gecode): `https://www.minizinc.org/`
   - Google OR-Tools CP-SAT Solver: `https://developers.google.com/optimization/cp/cp_solver`
   - Choco: `https://choco-solver.org/`
3. (20 points, provided that part 2 is completed) Evaluate your CSP model experimentally by 15 Slant puzzle instances of 3 difficulty levels (i.e., 5 easy, 5 normal, 5 difficult instances).[3]

   Summarize the results of your experiments in a table that shows, for each puzzle instance, the number of variables, the number of constraints, the number of backtracks/choices, and the CPU time.

   Discuss these results: What do you observe about the scalability of your CSP-based method for solving the Slant puzzle?

Submit the following two files at SUCourse+:

- A pdf file (at most 2 pages) containing
  - a description of your CSP model of the Slant puzzle, and
  - a discussion on whether A* or CSP is more appropriate for solving this puzzle.
- A zip file containing
  - your model of the Slant puzzle in the language of the CP solver,
  - three puzzle instances of different difficulty levels, and
  - the solutions of these instances computed by the CP solver.

In each deliverable above, please include your name and student id.

Demos You are expected to make a demo of your CSP-method so that we can grade parts 2 and 3 of the assignment. The demos are planned for the week following the deadline and will be scheduled later on.

[3]Note that you need to clarify how you define the difficulty level: How do you describe an easy instance? A normal instance? A difficult instance? It is preferable to present one example for each difficulty level in your report, to better explain your formulation of difficulty levels. In the demos, you are expected to show one example for each difficulty level.