**German University in Cairo**
**Faculty of Media Engineering and Technology**
**Dr. Nada Sharaf**

**Concepts of Programming Languages**, Spring 2019
**The Helsinki Puzzle**
**Deadline: 22/3/2019**

# Project Description

Given a square grid of size N, where the horizontal rows are numbered 1 to N from top to bottom and the vertical columns are numbered 1 to N from left to right. You must place a number in each cell of the N by N grid such that :-

- Each row is unique.
- Each row is exactly equal to one of the columns, however, it must **not** be the column with the same index as the row.
- If X is the largest number you place in the grid, then you must also place 1,2,...,X-1, where the condition $X \leq N$ is satisfied.

# Examples

For a $3 \times 3$ grid, you may have the following matrix

| | c1 | c2 | c3 |
|---|---|---|---|
| r1 | 2 | 1 | 2 |
| r2 | 2 | 2 | 1 |
| r3 | 1 | 2 | 2 |

defined by the following equalities $\begin{bmatrix} c1 & = & r2 \\ c2 & = & r3 \\ c3 & = & r1 \end{bmatrix}$

For a $4 \times 4$ grid, you may have the following matrix

| | c1 | c2 | c3 | c4 |
|---|---|---|---|---|
| r1 | 1 | 2 | 3 | 1 |
| r2 | 3 | 4 | 4 | 2 |
| r3 | 2 | 4 | 4 | 3 |
| r4 | 1 | 3 | 2 | 1 |

defined by the following equalities $\begin{bmatrix} c1 & = & r4 \\ c2 & = & r3 \\ c3 & = & r2 \\ c4 & = & r1 \end{bmatrix}$

**German University in Cairo**
**Faculty of Media Engineering and Technology**
**Dr. Nada Sharaf**

# Predicates to be added

You are going to solve this puzzle purely through Prolog. This means that you are **not allowed to use any clpfd libaries**. Your solution must utilize both techniques, unification and generate-and-test. You **should implement all of the following predicates.**

## grid_build/2

The predicate `grid_build(N,M)` should succeed only if `M` represents a grid that is N by N such that each cell in `M` contains an unbound variable.
Hint: `length(L,3)` produces a list of 3 unbound variables.

## grid_gen/2

The predicate `grid_gen(N,M)` should succeed only if `M` represents a grid that is N by N such that each cell in `M` contains a value from the valid range `1 .. N`.

## num_gen/3

The predicate `num_gen(F,L,R)` should succeed only if `R` represents a list of consecutive numbers starting from `F` until `L`.
Hint: `numGen(1,3,R)` succeeds when R = [1,2,3].

## check_num_grid/1

The predicate `check_num_grid(G)` succeeds if `G` does not contain a number `X` unless all the numbers `1 .. X-1` are there.

## acceptable_permutation/2

The predicate `acceptable_permutation(L,R)` should succeed only if `R` represents an acceptable permutation of the list `L`.
Hint: [2,1,3] is not an acceptable permutation of the list [1,2,3] because 3 did not change it's position.

**German University in Cairo**
**Faculty of Media Engineering and Technology**
**Dr. Nada Sharaf**

## acceptable_distribution/1

The predicate `acceptable_distribution(G)` should succeed only if no row is placed in a column with the same index and no column is placed in a row with the same index.

## row_col_match/1

The predicate `row_col_match(M)` should succeed only if each row is equal to a column with a different index and each column is equal to a row with a different index.

## trans/2

The predicate `trans(M,M1)` should succeed only if `M1` represents a transposed version of the matrix `M`.
Hint: This needs to be approached through rows and columns.

## distinct_rows/1

The predicate `distinct_rows(M)` should succeed only if `M` represents a matrix `M` where all rows are unique.

## distinct_columns/1

The predicate `distinct_columns(M)` should succeed only if `M` represents a matrix `M` where all columns are unique.

## helsinki/2

The predicate `helsinki(N,G)` should succeed only if `G` is a square grid of size `N*N` that satisfies all the helsinki puzzle properties.