COL 765 **November 3, 2021**

Assignment 3

Instructor: Subodh Sharma **Due: 23:59 hrs November 14, 2021**

Problem 1:

1. (Query) Assume we have a data with a binary predicate `parent`, which is true of two people *iff* the person specified as the first argument is the parent of the person specified as the second argument. Write a query that defines the property of being childless. (Hint: try using the aggregate operator `countofall`). Use the kinship dataset of question (2).

2. (update) Suppose we have a kinship dataset with a binary predicate `parent` and a unary predicate `male`. Write update rules to *replace* all factoids using the `parent` predicate with equivalent factoids using the binary predicates `father` and `mother`. Kinship Dataset:

```
parent(arti, babli)
parent(arti, bunty)
parent(babli, chitra)
parent(babli, chintan)
parent(bunty, divya)
parent(bunty, divesh)
male(bunty)
male(chintan)
male(divesh)
```

3. (recursion) Consider the classic puzzle of towers of Hanoi. The recursive formulation of this problem is straightforward. Write a Prolog program that solves this puzzle. Assume that there are only three pegs `left, centre`, and `right`. The discs are numbered such that if $i < j$ then disc j is of larger size that disc i. The output of a query to the Prolog program should print in order a sequence of moves in the following form: "Move disc 1 from left to centre".

4. (subgoal ordering, constraint satsifaction) The *cryptarithmetic problem* is characterized by a finite set of letters and a finite set of numbers and an arithmetic constraint written in terms of the letters. Consider a square of order 3 – (A,B,C), (D, E, F), (G,H,I); here each row of the square is shown as a tuple enclosed in parenthesis. Write a Prolog program that maps the digits to a non-zero number in the set $\{0, \ldots, 9\}$ s.t. the each row, columm and diagonals add up to the same number. The output of the program should list the mapping of digits to numbers.

5. (Knowledge representation, Side-effects, backtracking) The N-queens problem is a challenge to set N queens on a NxN grid so that no queen can attack any other queen. Create a Prolog program to solve the N-queens problem. The output of the program should be a list of (row,columm) positions where the N (for some concrete value of N) queens can be placed.

Problem 2: Extra Credit

Develop a Rikudo puzzle solver in Prolog. The puzzle Rikudo was created in 2015. Check this website to enjoy the puzzle and get a clearer idea about the rules.

- The hexagonal cells (empty or filled) tile a large hexagon.
- The number of cells n can be 37, 61 or 91 depending upon the size.
- The numbers in empty cells should be filled to create a hamiltonian path of consecutive numbers from 1 to $n - 1$.
- The centre cell cannot contain a number, hence, no path can go through it.
- The numbers cannot be repeated.
- In some cases, a link will be given. The link indicates that the path is crossing though that edge i.e., if cell A and B are linked they should contain the consecutive numbers making a path from A to B or B to A.
- A correct solution must fill all the cells each having a unique number.

A Rikudo puzzle is defined in terms of:
Size An integer n denoting size of the grid. It can take values 37, 61 or 91.
Pre-filled Numbers A list of tuples (x, y, k) which shows that cell (x, y) has been pre-filled with the number k.
Links A list of tuples will be given denoting the links between neighbours.
Result The output is a list of 3-tuples of the form (x, y, k), consistent with pre-filled numbers and the links. All the numbers in the result should be from the range of allowed numbers with no repetitions. The center cell (cell (0, 0)) should contain the number -10.

Create a function `rikudo` to solve the puzzle. The function should take three arguments, namely *Size*, *Pre-filled* and *Links*, and return *Result* in the format defined above.

Submission Instructions

1. A single zip file `entry-number.zip` should be submitted on Moodle.
2. The zip file should contain exactly one file per question. The solution files can be named in the following way: for question 1, name the file `entry-number-q1.pl` and so on.