CSE4102: Assignment #8 **Spring 2017**

Logic Programming

Date: 04/13/2017
Due: 04/20/2017

---

1. [10] Family Tree

   Your first task is to play with a family tree program (`family.pl`) and define a few predicates that capture common family relations. Feel free to assume the existence of a collection of axioms (prolog facts) to describe a group of individuals (gender, and father/mother relations). Given this small database (feel free to use the same as in class or to type in your own family tree!) answer the following

   - `brother(X,Y)` implements a predicate that holds if X is the brother of Y.
   - `sister(X,Y)` implements a predicate that holds if X is the sister of Y.
   - `uncle(X,Y)` implements a predicate that holds if X is the uncle of Y.
   - `aunt(X,Y)` implements a predicate that holds if X is the aunt of Y.
   - `cousin(X,Y)` implements a predicate that holds if X is a cousin of Y.

2. [10] Simple numerical computations

   Write in `power.pl` a prolog predicate that computes the n^{th} power of a base x, i.e., it computes x^n. The predicate is ternary and can be used as follows

   ```
   power(5,2,P), format('P␣is␣~d\n',[P]).
   ```

   Clearly, the `format` predicate is used to format (like `printf` in C) arguments to the standard output. Check the documentation `http://www.gprolog.org/manual/html_node/gprolog038.html#format%2F3` for details. Clearly, that predicate will not be multi-directional.

3. [10] Burn the Witch!

   Consider the dialog from Monty Python `http://www.montypython.net/scripts/HG-witchscene.php` in which Sir Bedevere develops a logical argument to answer the question: "Is the girl a witch?" Your task is to encode this rationale as a prolog program `witch.pl`. Naturally, you will need axioms to capture the gender of the 'girl' as well as how her weight compares to that of the proverbial duck. You will also need a few predicates for the remainder of the logical argument. Your ultimate objective is to produce a predicate `witch(X)` and have prolog answer the question for you to determine whether the girl is, or is not, a witch. If you don't recall the scene, take a minute and head over to youtube to watch the scene again! `https://www.youtube.com/watch?v=zrzMhU_4m-g`.

4. [10] A Generator

   Write in `gen.pl` a prolog predicate which, upon backtrack, will produce a sequence of increasing natural numbers starting from some seed S. Namely,

   ```
   gen(1,N)
   ```

   is the invocation of your `gen/2` predicate, that will produce the stream of values $1, 2, 3, 4, 5, \cdots$ with each new value produced on backtrack. (Namely, at the prompt, hitting ';' will ask prolog to produce the next value.

5. [10] The Farmer, the wolf, the goat and the cabbage

This question requires lists that we will cover on Tuesday. So while you can start thinking about this classic puzzle, you won't have the elements to answer it before Tuesday afternoon.

A farmer and his goat, wolf, and cabbage come to the East bank of a river that they wish to cross. There is a boat, but it has only room for two, and the farmer is the only one who can row. If the goat and the cabbage are on the same bank with no supervision from the farmer, the goat will eat the cabbage. Similarly, if the wolf and the goat are together without the farmer, the goat will be eaten. Devise a series of crossings of the river so that all concerned parties make it across safely to the opposite bank.

The prolog program should have a predicate `solve(L)` that enumerate the ways to get across safely from the East to the West bank of the river. The list L is a list of moves. Note that since there are at most two spots in the boat, the farmer could be crossing alone, but it is also possible for him to cross with either the wolf, the goat or the cabbage. A crossing only makes sense if everyone is safe (there is no risk for the goat or the cabbage to get eaten).

As a hint, consider that you should produce the crossing plans in order of increasing length (shorter plans should be given first. A possible session where the source code in 'river.pl' is loaded and queried is shown below

```
src (master) $ gprolog
GNU Prolog 1.4.4 (64 bits)
Compiled Apr 22 2014, 10:06:54 with clang
By Daniel Diaz
Copyright (C) 1999-2013 Daniel Diaz
| ?- consult('river.pl').
compiling /Users/ldm/Documents/courses/CSE4102/src/river.pl for byte code...
/Users/ldm/Documents/courses/CSE4102/src/river.pl compiled,
24 lines read - 7567 bytes written, 6 ms

(2 ms) yes
| ?- solve(L).

L = [move(fg,ew),move(f,we),move(fw,ew),move(fg,we),move(fc,ew),move(f,we),move(fg,ew)] ? ;

L = [move(fg,ew),move(f,we),move(fc,ew),move(fg,we),move(fw,ew),move(f,we),move(fg,ew)] ? ;

L = [move(fg,ew),move(fg,we),move(fg,ew),move(f,we),move(fw,ew),move(fg,we),move(fc,ew),
     move(f,we),move(fg,ew)] ?

(3 ms) yes
| ?-
```

In this trace, a move such as `move(fg,ew)` states that the farmer (f) and the goat (g) are both moving in the boat from east to west. In this session, the user asks for the first three solutions. To ease grading, please consider using the same notation.

Have fun!

PS/to handin, zip all your prolog files in an archive `prolog.zip` and submit that.