1: Starting Prolog

Open Prolog by calling `gprolog` in a terminal window. If you prefer, you can open a shell in an Emacs window by typing <esc x shell> into Emacs.

Construct your knowledge base in a file in your favourite editor and load it by typing `consult('kb_name.pl').`

To enter knowledge bases directly into Prolog, type [user] at the Prolog prompt. Prolog now expects input from you. You can cut and paste your knowledge base. End your input by typing <ctrl-d>. Prolog will now go back into query mode.

2: Lecture Examples

Download kb1.pl and printlist.pl from Moodle, load them into your interpreter and play around. Ask yes/no questions, try queries with variables and wrap your head around recursion (hint: use `trace`).

3: Append in Prolog

In Prolog, one list (written as [a,b,c] or [] for the empty list) is appended to another by the following code:

```
acc_append([], Ys, Ys).
acc_append([X|Xs], Ys, [X|Zs]) :- acc_append(Xs, Ys, Zs).
```

Analyse the code by answering the following questions. You may use `trace`.

What is the base case for recursion?

Which of the variables accumulates the result?

What value should this variable therefore have for the initial call?

What happens when the non-base case rule is applied? Where is the new call with the smaller argument that allows recursion to terminate?

4: Optional: Logic puzzle (from Learn Prolog Now! with thanks)

There is a street with three neighbouring houses that all have a different colour, namely red, blue, and green. People of different nationalities live in the different houses and they all have a different pet. Here are some more facts about them:

- The Englishman lives in the red house.
- The jaguar is the pet of the Spanish family.
- The Japanese lives in the blue house.
- The snail keeper lives to the left of the blue house.

Who keeps the zebra? Define a predicate zebra/1 that tells you the nationality of the owner of the zebra!

Hint 1: Think of a representation for the houses and the street. Code the constraints in Prolog.

Hint 2: member/2 checks whether a term ist he member of a given list.

Hint 3: is/2 does arithmetic computations – e.g., is(X,7+2).