Getting started with Prolog

We have a public domain implementation of Prolog called Gnu Prolog. It should be on your search path typing

```prolog
  gprolog
```

should start it.

Gnu Prolog is Unix platform software. For the PC you can down load SWI Prolog or a trial version of ALS Prolog which has a nice interface. Both these last two support modules which GNU Prolog does not.
Constants, Variables, predicates, and functions

- Constants – begin with a lower case letter or a in single quotes.

- Variables – begin with an upper case letter or _

- Predicates are relations. Predicate names are constants.

- A function can be expressed as a predicate thus square(X,Y) to represent the relation (function) Y is X squared.
First Steps: data base programming

Copy the data given in Worksheet 3 (p14) of Clocksin into a file counties.pl. Later we will add the rules.

\[
\begin{align*}
\text{border(sussex, kent)}. & \quad \text{border(sussex, surrey)}. \\
\text{border(surrey, kent)}. & \quad \text{border(hampshire, sussex)}. \\
\text{border(hampshire, surrey)}. & \quad \text{border(hampshire, berkshire)}. \\
\text{border(berkshire, surrey)}. & \quad \text{border(wiltshire, hampshire)}. \\
\text{border(wiltshire, berkshire)}. & \\
\end{align*}
\]

Note: The periods are required. No space between border and the opening parenthesis.

Start gprolog, and at the ‘?-’ prompt type [counties].

The goal halt. exits Prolog.
Simple Database programming

Having loaded the database counties.pl into the system. We can now try some *queries*.

```
| ?- border(sussex, kent).
true ?
yes
| ?- border(sussex, hampshire).
no
| ?- border(X, sussex).
X = hampshire ?
no
| ?- border(sussex, Y).
Y = kent ?  % I entered a ; here
Y = surrey
yes
```

Note that the queries are capable of *instantiating* variables.
We can do more complex queries, here, is interpreted as ‘and’

| ?- border(wiltshire, X), border(X, Y).

X = hampshire
Y = sussex ?

X = hampshire
Y = surrey ?

X = berkshire
Y = surrey

yes
But have we captured adjacency?

But recall that we had

\[ \text{?– border(sussex, hampshire).} \]
\[ \text{no} \]

Even though we have the fact border(hampshire, sussex).

So the system doesn’t know that border is symmetric. We can use a simple rule to add this to the system. Rules have the form

\[ \text{head :- body.} \]

head is a single term, body is (for the moment) a conjunction (using ,) of terms.

\[ \text{adjacent(X,Y) :- border(X,Y).} \]
\[ \text{adjacent(X,Y) :- border(Y,X).} \]
Make a new file *counties1.pl* that contains this rule as well as the original facts. The two *clauses* serve as alternate ways of satisfying the goal *adjacent*.

Now we have:

?- adjacent(sussex, hampshire).

true ?
Another rule and a new problem

If we add to *counties1.pl* the rule:

```prolog
affordable(X,Z) :-
    adjacent(X,Y),
    adjacent(Y,Z).
```

Now let’s try some queries.

?- affordable(wiltshire, sussex).

true ?

yes

| ?- affordable(wiltshire, kent).

no

But we also find affordable(kent,kent), do we want this?
Lists

• \([\ ]\) – The null list;

• \([a,b,c,d]\) – an explicit list;

• \([H|T]\) – the list whose head is \(H\) and with tail \(T\)

• \('.'(H,T) = [H|T]
Member

Actually member is almost always a built-in, but of it isn’t here is memberp/2.

\[
\text{memberp}(X, \ [X|T]) . \\
\text{memberp}(X, \ [H|T]) :- \\
\quad \text{memberp}(X, \ T) .
\]

Anonymous (or don’t care) variables are ones that only appear once in a rule. _ or \_varname can be used for these. Thus we would write \text{memberp}(X, \ [X|\_]). since we don’t care about the tail.
Arithmetic

Arithmetic is done using the predicate \texttt{is/2}. Thus

$$X \text{ is } (3 + 4) \times 5.$$ 

Succeeds setting \(X\) to 35.

Note that the right hand side must be completely instantiated when the goal is tried.

We have the following built-in \textit{arithmetic} predicates

\texttt{==} (equal) note that \(3 == 1 + 2\). succeeds where as \(3 = 1 + 2\). fails. Why?

\texttt{>, >=, <, <=} with the obvious meanings.