MA410 Prolog Practical 2 - Family Tree

1 Ready-made family tree

(a) Download the files `family.pl` and load it into prolog by using the `consult` command.

(b) Now try to find all the children of `carmel` by

```
?- parent(carmel, X).
```

and repeatedly pressing `;`.

Find out who is her husband by

```
?- married_to(carmel, X).
```

(c) Edit the file `lab2rels.pl` and add definitions for `mother` and `father`:

```
mother(X, Y) :- parent(X, Y), female(X).
father(X, Y) :- parent(X, Y), male(X).
```

Test your program, e.g. who are the `father` and `mother` of `diarmuid`?

(d) Experiment with the operators `=`, `==`, `\==`, `\=` within prolog. Try out a few examples, e.g.

```
?- a = b.    ?- X = a.
?- X \= a.   ?- X \== a.
```

(e) Create predicate `diff(X, Y)` as below.

```
diff(X, Y) :- X \== Y.
```

(f) Define the following predicates:

- `sibling(X,Y)`, `brother(X,Y)`, `sister(X,Y)`
- `grandparent(X, Y)` (`X` the grandparent of `Y`)
- `grandmother(X, Y)` (`X` the grandmother of `Y`)
- `grandfather(X, Y)` (`X` the grandfather of `Y`)
- `bl_uncle(X, Y)` (`X` a blood-related uncle of `Y`)
- `uncle_in_law(X, Y)` (`X` an uncle-in-law of `Y`)
- `uncle(X, Y)` (`X` is any type of uncle of `Y`)

Hint: figure it out on paper first, e.g. a sibling of x is someone not x who shares the same parent as x.

(g) Use predicates above to check relationships e.g. who are david's siblings, grandparents?

(h) **Task:** Create definitions for predicates:
`aunt(X,Y)`, `firstcousin(X,Y)`, `ancestor(X,Y)`, `descendant(X,Y)`.
Create sample input & (non-empty) output for each case & e-mail the code/results with subject heading "FT Lab - Relations". To find a legitimate input, look for appropriate family relationships in the file.

2 Creating our own family tree

Task: Download the file `myfam.pl` and create your own family tree based on `family.pl`. If you prefer, you can create a fictitious family tree. E-mail on this file together with a few test inputs/outputs with subject heading "FT Lab - My Family Tree". In order to test relations in your family tree, you'll need to change the include file in `lab2rels.pl` to point to `myfam.pl`.

3 A.I. based questions

Our aim is to allow the user to ask questions in a more natural way, e.g. "`who are the parents of X.`", etc.

We'll use the predicate `setof` in order to organise sets of a particular type. To see how it works, check out `http://www.cse.unsw.edu.au/~billw/prologdict.html`.

3.1 First Example

(a) Create a predicate `sister_list` as follows:

```
sister_list(X, L) :- setof(Y, sister(Y,X), L).
```

(b) We create a series of `:- op` statements:

```
:- op(700, xfy, are).
:- op(600, fx, the).
:- op(500, fx, sisters).
:- op(400, fx, of).
```

(c) We now define:

```
Who are the sisters of X :- sister_list(X, Who).
```

(d) Test out the command with various inputs, e.g.

```
Who are the sisters of daniel.
```

(e) Now add code so that the user can ask

```
Who are the brothers of X.
```

(f) Alter the code to allow the user enter in

```
Give me the brothers of X.
```

3.2 Improving the method

In the next lab, we will try get the computer to reply in a more natural way, e.g. if we ask "`who are the brothers of tom`", it might reply "`john, pat and harry.`" as opposed to saying `Who = [john, ...]`.

Also, we would like a more generalised approach to allowing a user to ask questions, i.e. "who are the `X` of `Y`" where `X` might be replaced by `brothers`, `sisters`, `cousins`, etc.