Programming Paradigms — Prolog Lab sheet

For the Prolog part of the course we are using SWI-Prolog but you can use another "Edinburgh" Prolog implementation if you wish.

Preamble

```
% swipl
Welcome to SWI-Prolog (Multi-threaded, 64 bits, Version 6.2.1)
Copyright (c) 1990-2012 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

?-  halt.
%
```

1. Assume given a set of facts of the form `father(name1,name2)` (`name1` is the father of `name2`).
   (a) Define a predicate `brother(X,Y)` which holds iff `X` and `Y` are brothers.
   (b) Define a predicate `cousin(X,Y)` which holds iff `X` and `Y` are cousins.
   (c) Define a predicate `grandson(X,Y)` which holds iff `X` is a grandson of `Y`.
   (d) Define a predicate `descendent(X,Y)` which holds iff `X` is a descendent of `Y`.
   (e) Consider the following genealogical tree:

   ```
   father(a,b).
   father(a,c).
   father(b,d).
   father(b,e).
   father(c,f).
   ```

   whose graphical representation is:

   ```
         a
        / \
       b   c
      / \  |
     d   e f
   ```

   Say which answers, and in which order, are generated by your definitions for the queries

   ```
   ?- brother(X,Y).
   ?- cousin(X,Y).
   ?- grandson(X,Y).
   ?- descendent(X,Y).
   ```

2. Define a predicate `reverse(L,K)` which holds if and only if the list `K` is the reverse of the list `L`.

3. (a) Create a basic Prolog knowledge base (consisting of facts) describing the following relationships on Twitter:

   Anne follows Fred

   Fred follows Julie and Susan

   John follows Fred

   Julie follows Fred

   Susan follows John and Julie

   (b) Add some for facts describing that the persons above tweeted the following messages:

   Anne tweeted tweet1 and tweet5

   Fred tweeted tweet2, tweet7, and tweet8

   John tweeted tweet3, and tweet4

   Julie tweeted tweet6

   Susan tweeted tweet9 and tweet10

   (c) Ask your knowledge base the following questions (you may want to add rules to the knowledge base to help you answer the questions):

      i. Which tweets can Fred see (assuming that only direct followers will see a tweet)?
      ii. Find all the persons who are friends, i.e., they follow each other.
      iii. Output for each person which tweets they can see.
      iv. If everyone retweets every tweet they get, which tweets can Fred see now?

4. Define a predicate `add_up_list(L,K)` which, given a list of integers `L`, returns a list of integers in which each element is the sum of all the elements in `L` up to the same position. Example:

```
?- add_up_list([1,2,3,4],K).
   K = [1,3,6,10];
   no
```

5. (a) Create a knowledge base (consisting of facts and rules) describing the following:

   Malcom is Scottish and plays rugby.

   Claude is French and plays football.

   John is British and plays cricket.

   Owen is from Wales and plays chess.

   Sean is from Nortern Ireland, but doesn't play anything.

   Nigel is English and plays football.

Someone is British, if they are English, Northern Irish, Scottish, or Welsh.

Someone is a sportsman if he plays cricket, football, or rugby.

(b) Now answer the following questions using your knowledge base

i. Is Owen a sportsman?

ii. List all the British sportsmen.

iii. List all the nationalities of the football players.

6. Define a predicate `merge(L,K,M)` which, given two ordered lists of integers `L` and `K`, returns an ordered list `M` containing all the elements of `L` and `K`.

7. Write a Prolog program that computes the greatest common divisor (*gcd*) of two numbers A and B. If $A \leq B$, then $gcd(A, B) = gcd(A, B - A)$. $gcd(A, 0)$ is equal to A.

8. [Difficult]

Consider a representation of binary trees as terms, as follows

`emptybt` the empty binary tree

`consbt(N,T1,T2)` the binary tree with root `N` and left and right subtrees `T1` and `T2`

(a) Define a predicate `preorder(T,L)` which holds iff `L` is the list of nodes produced by the preorder traversal of the binary tree `T`.

(b) Define a predicate `search_tree(L,T)` which, given a list of integers `L`, returns a balanced search-tree `T` containing the elements of `L`.

Credits

This worksheet is a combination of questions by Sven Helmer, Ivan Bratko, myself, and myriad others who I have forgotten over the years (apologies).