PROLOG-List-Operations-1 – Documentation

Csc 600-01

Keawa Rozet

Code available at: <https://github.com/krozet/PROLOG-List-Operations-1>

Prompt

Write a PROLOG program that investigates family relationships using lists. The facts should be organized as follows:

m([first\_male\_name, second\_male\_name, …, last\_male\_name]).

m([first\_female\_name, second\_female\_name, …, last\_female\_name]).

family([father, mother, [child\_1, child\_2, …, child\_n]]).

Write rules that define the following relations:

male(X)

female(X)

father, mother, parent

siblings

brother, brothers

sister, sisters

cousins

uncle, aunt

grandchild, grandson, granddaughter

greatgrandparent

ancestor

For each of these rules show an example of its use.

What my code does

In my code I have included 4 generations worth of family to test. In conjunction with the nature of PROLOG, my code can tests rules either by comparing two values or by displaying all the values that satisfy the rule with a value provided by the user. I tried to show case this in my code example.

For the examples I use the same names a lot for the sake of readability while grading, but a number of different value combinations will work for each rule.

Code

m([keawa, kalani, haku, alii, kaeo, eric, shawn, wayne, steve,

brad, caleb, kai, mark, roger, mako, keina, ben, mathew, joel,

kainalu, tim, sam, richard, kamehameha]).

f([elizabeth, raeshell, sarah, geri, jordan, zeli, hannah, mia,

haley, lucy, cici, kani, buni, sheryl, opau, marybeth, heather,

chloe, paina]).

%fourth generation

family([eric, raeshell, [keawa, kalani, elizabeth]]).

family([shawn, sarah, [alii, kaeo, jordan]]).

family([steve, hannah, [haley, brad, caleb]]).

%third generation

family([wayne, geri, [raeshell, sarah, steve]]).

family([kai, zeli, [eric, mia]]).

%second generation

family([mark, lucy, [wayne, cici]]).

family([ben, sheryl, [geri, mathew, joel]]).

family([roger, kani, [zeli, buni, mako, keina]]).

family([kainalu, opau, [kai]]).

%first generation

family([tim, marybeth, [lucy]]).

family([sam, heather, [ben]]).

family([richard, chloe, [roger]]).

family([kamehameha, paina, [kainalu]]).

male(X) :- m(M), member(X, M).

female(X) :- f(F), member(X, F).

father(X) :- male(X), parent(X).

father(X, Y) :-

family([M, F | [Children]]),

(X = M; X = F), male(X),

member(Y, Children).

mother(X) :- female(X), parent(X).

mother(X, Y) :-

family([M, F | [Children]]),

(X = M; X = F), female(X),

member(Y, Children).

parent(X) :- family([M, F | [\_]]), (X = M; X = F).

parent(X, Y) :-

family([M, F | [Children]]),

(X = M; X = F),

member(Y, Children).

siblings(X, Y) :-

family([\_, \_ | [Children]]),

member(X, Children),

member(Y, Children),

X \= Y.

brothers(X, Y) :-

siblings(X, Y),

male(X),

male(Y).

brother(X, Y) :-

siblings(X, Y),

male(X).

sisters(X, Y) :-

siblings(X, Y),

female(X),

female(Y).

sister(X, Y) :-

siblings(X, Y),

female(X).

cousins(X, Y) :-

family([M, \_ | [\_]]),

parent(M, X),

siblings(M, S),

parent(S, Y).

cousins(X, Y) :-

family([\_, F | [\_]]),

parent(F, X),

siblings(F, S),

parent(S, Y).

uncle(X, Y) :-

family([M, \_ | [\_]]),

parent(M, Y),

brother(X, M).

uncle(X, Y) :-

family([\_, F | [\_]]),

parent(F, Y),

brother(X, F).

aunt(X, Y) :-

family([M, \_ | [\_]]),

parent(M, Y),

sister(X, M).

aunt(X, Y) :-

family([\_, F | [\_]]),

parent(F, Y),

sister(X, F).

grandchild(X, Y) :-

parent(M, X),

parent(Y, M).

grandson(X, Y) :-

grandchild(X, Y),

male(X).

granddaughter(X, Y) :-

grandchild(X, Y),

female(X).

greatgrandparent(X, Y) :-

grandchild(Y, M),

parent(X, M).

ancestor(X, Y) :-

greatgrandparent(M, Y),

parent(X, M).

show\_families :- family(F), write\_list(F).

write\_list([X,Y | [Sub]]) :-

write(X), nl, write(Y), nl,

write\_sublist(Sub).

write\_sublist([]).

write\_sublist([H|T]) :-

write(H), nl,

write\_sublist(T).

Example

> swipl

Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)

SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.

Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org

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

?- [listops].

true.

?- male(keawa).

true .

?- female(X).

X = elizabeth ;

X = raeshell ;

X = sarah .

?- father(eric, keawa).

true .

?- mother(zeli, eric).

true .

?- parent(X, keawa).

X = eric ;

X = raeshell ;

false.

?- siblings(brad, X).

X = haley ;

X = caleb ;

false.

?- brother(mako, keina).

true .

?- sister(haley, brad).

true .

?- sisters(raeshell, sarah).

true .

?- cousins(keawa, X).

X = alii ;

X = kaeo ;

X = jordan ;

X = haley ;

X = brad ;

X = caleb ;

false.

?- uncle(steve, keawa).

true .

?- aunt(X, keawa).

X = mia ;

X = sarah ;

false.

?- grandchild(keawa, wayne).

true .

?- grandson(steve, mark).

true .

?- granddaughter(haley, wayne).

true .

?- greatgrandparent(kainalu, keawa).

true .

?- ancestor(kamehameha, keawa).

true .