

Family Relationships

john, jim, and jerry are males, while sue, carol, and alice are females.

jerry and carol are john and alice's offspring, ted is jerry and sue's offspring, and bob is jim and carol's offspring.

- 1) male(john).
- 2) male(jim).
- 3) male(jerry).
- 4) female(sue).
- 5) female(carol).
- 6) female(alice).
- 7) parent(john, jerry).
- 8) parent(john, carol).
- 9) parent(alice, jerry).
- 10) parent(alice, carol).
- 11) parent(sue, ted).
- 12) parent(jerry, ted).
- 13) parent(carol, bob).
- 14) parent(jim, bob).

We store the 14 assertions in the file “family.pro”.

pl

Welcome to SWI-Prolog (Version 3.1.2)

Copyright (c) 1993-1998 University of Amsterdam. All rights reserved.

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

?- consult('family.pro'). (or ['family.pro'].)
< message, possible errors, etc. >

yes

?- listing.
< lists all rules in the system >

?- listing(parent).
< lists all parent rules in the system >

?- parent(sue, ted).

yes

?- parent(alice, X).
X = jerry < waits for response, ; means more >
X = carol < ; >

no

?- parent(Y, carol).
Y = john < ; >
Y = alice < ; >

no

?- male(X).

X = john ;

X = jim <c-r>

yes

?- male(bob)

no

?-

1) male(john).

2) male(jim).

3) male(jerry).

4) female(sue).

5) female(carol).

6) female(alice).

7) parent(john, jerry).

8) parent(john, carol).

9) parent(alice, jerry).

10) parent(alice, carol).

11) parent(sue, ted).

12) parent(jerry, ted).

13) parent(carol, bob).

14) parent(jim, bob).

15) sib(X, Y) :- parent(Z, X),
parent(Z, Y),
X \== Y.

16) brother(X, Y) :- sib(X, Y), male(X)

15) sib(X, Y) :- parent(Z, X),
parent(Z, Y),
X \== Y.

16) brother(X, Y) :- sib(X, Y), male(X)

Let's try to solve the goal

?- brother(jerry, carol). RULE 16
 {jerry/X, carol/Y}

?- sib(jerry, carol), RULE 15
 male(jerry). {jerry/X, carol/Y}

?- parent(Z, jerry), RULE 7 {john/Z}
 parent(Z, carol), jerry \== carol,
 male(jerry).

?- parent(john, carol), RULE 8
 jerry \== carol, male(jerry).

?- jerry \== carol, male(jerry). \==, RULE 3

yes DONE!!

- 3) male(jerry).
- 7) parent(john, jerry).
- 8) parent(john, carol).
- 15) sib(X, Y) :- parent(Z, X), parent(Z, Y), X \== Y.
- 16) brother(X, Y) :- sib(X, Y), male(X)

How would Prolog do it?

?- spy(brother).

Spy point on brother/2

Yes

[debug] ?- brother(jerry,carol).

* Call: (7) brother(jerry, carol) ? creep

Call: (8) sib(jerry, carol) ? creep

Call: (9) parent(_L151, jerry) ? creep

Exit: (9) parent(john, jerry) ? creep

Call: (9) parent(john, carol) ? creep

Exit: (9) parent(john, carol) ? creep

Call: (9) jerry \== carol ? creep

Exit: (9) jerry \== carol ? creep

Exit: (8) sib(jerry, carol) ? creep

Call: (8) male(jerry) ? creep

Exit: (8) male(jerry) ? creep

* Exit: (7) brother(jerry, carol) ? creep

Yes

[debug] ?- abort.

Execution Aborted

?-