Praktikum 3

Eesmärk: Rekursiooni kasutamine sügavuti otsingul

Eelmises praktikumis tuli luua sugulaste teadmusbaas kasutades predikaate: mother/2, married/2, male/1, female/1 ja lisada reeglid sugulaste leidmiseks teadmusbaasist.

Näiteks:

```
% Seost, kes on kelle ema, esitasime faktiga: mother(Child, Mother).
mother('Mai', 'Kai').
mother('Kai', 'Kadri').
mother('Kadri', 'Epp').
mother('Epp', 'Leida').

% vanaema leidsime reegiga: grandmother(Child, Grandmother).
grandmother(Child, Grandmother):-
        mother(Child, Mother),
        mother(Mother, Grandmother).
```

```
?- grandmother('Mai', 'Kadri').          % Kas Kadri on Mai vanaema?
true.

?- grandmother('Kai', Vanaema).          % Kes on Kai vanaema?
Vanaema = 'Epp'.
```

Kui me sooviksime teada vana-vanaema, siis tuleks kirjutada reegel:

```
grand_grandmother(Child, Grandgrandmother):-
        mother(Child, Mother_1),
        mother(Mother_1, Mother_2),
        mother(Mother_2, Grandgrandmother).
```

```
?- grand_grandmother('Mai', 'Epp').      % Kas Epp on Mai vana-vanaema?
True.
```

Rekursioon

```
ancestor(C, G) :- mother(C, G).
ancestor(C, G) :-
        mother(C, Mother),
        ancestor(Mother, G).
```

```
?- ancestor('Mai', 'Epp').      % Kas Epp on Mai esiema?
True.

?- ancestor('Mai', Vanem).      % Kes on Mai esiemad?
Vanem = kai ;
Vanem = kadri ;
Vanem = epp ;
Vanem = leida ;
false.
```

Rekursiooni kirjutamiseks on vaja vähemalt 2-te reeglit:

- baasreegel (reegel, mis peatab rekursiooni)
- ja teine, mis sisaldab rekursiooni.

ancestor/2 reeglit rakendades saame teada, kas või kes on kellegi vanem.

Kui me sooviksime teada ka järgnevust e. kuidas on Leida Mai vana-vana-vanaema, siis saame kasutada järgnevat reeglit:

```
ancestor(X, Y, tee(X,Y)) :- mother(X, Y).
ancestor(X, Y, tee(X,Path)) :-
        mother(X, Z),
        ancestor(Z, Y, Path).
```

```
?- ancestor(mai, leida, Puu).
Puu = tee(mai, tee(kai, tee(kadri, tee(epp, leida)))) .
```

Täiendame emaks olemise fakti parameetriga, mis näitab kui vana oli ema lapse sünni ajal.

```
% mother(Laps, Ema, Ema_vanus).
mother(mai, kai, 24).
mother(kai, kadri, 35).
mother(kadri, epp, 27).
mother(epp, leida, 18).
```

Lisame reegili ancestor/4, mis lisaks eelnevale arvutab, milline on vanusevahe esivanemaga.

```
ancestor(X, Y, tee(X,Y), V) :- mother(X, Y, V).
ancestor(X, Y, tee(X, Z ,Path), V) :-
        mother(X, Z, Xv),
        ancestor(Z, Y, Path, Vz),
        V is Xv + Vz.
```

```
?- ancestor(mai, leida, Sugupuu, Vanus).
Sugupuu = tee(mai, kai, tee(kai, kadri, tee(kadri, epp, tee(epp, leida)))),
Vanus = 104 .
```

Ülesanne

1. Laienda oma sugulste teadmusbaasi nii, et seal oleks vähemalt neli sugupõlve.

2. Lisa teadmusbaasile rekursiivne reegel ancestor/2, mis leiab isiku kõik esivanemad.

```
?- ancestor(Child, Parent).
```

2. Lisa teadmusbaasile rekursiivne reegel male_ancestor/2, mis leiab meessoost esivanemad.

```
?- male_ancestor(Child, Parent).
```

3. Lisa teadmusbaasile rekursiivne reegel female_ancestor/2, mis leiab naissoost esivanemad.

```
?- female_ancestor(Child, Parent).
```

4. Lisa teadmusbaasile reegel ancestor/3, mis leiab N-nda sugupõlve esivanemad (nii mees- kui ka naissoost).

```
?- ancestor1(Child, Parent, N).
% NT. Kui N = 2, siis leitakse teise sugupõlve esivanemad
```

5. Lisa teadmusbaasile reegel ancestor2/3, mis leiab esivanemad, kellel on rohkem kui X last.

```
?- ancestor2(Child, Parent, X).
% NT. Kui X = 2, siis leitakse kõik esivanemad, kellel on rohkem kui X last
```