Heinrich Heine Universität Düsseldorf

Software Engineering and Programming Languages
Prof. Dr. Michael Leuschel
David Geleßus, Fabian Vu

Introduction to Logic Programming – WS 2023
Exercise Sheet 2

1 Exercises

The exercises will be discussed on 31st October 2023.

Exercise 1 (Lecture – Inverted Classroom)
Watch the lecture video *LP4 - Propositional Logic*[1] in the HHU Mediathek. The corresponding slides are uploaded in ILIAS: 3_PropLogic.pdf

The complete playlist is available at: `https://mediathek.hhu.de/playlist/691`.

Note: you have to log in with your HHU account (Uni-Kennung) to see the lecture videos!

Exercise 2 (Knowledge Base and Queries)
Implement the following facts in Prolog:

1. Siegfried loves Krimhild and likes Gunther.
2. Krimhild loves Siegfried and hates Brunhild.
3. Gunther loves Brunhild and likes Krimhild and Hagen.
4. Brunhild hates Siegfried, Gunther and Krimhild.
5. Hagen hates Siegfried and everyone who loves Siegfried.
6. Brunhild likes everyone who hates Siegfried.
7. Alberich hates everyone but himself.

Answer the following questions with Prolog queries:

1. Who does Brunhild like?
2. Who hates Siegfried?
3. Which couples[2] could be formed?

[1] `https://mediathek.hhu.de/watch/55da11bf-a3b0-4fcc-977f-8d9aaedfeae0`
[2] two persons that love each other ;-)

Exercise 3 (First-Order Logic)
Translate the following statements to First-Order Logic:

1. The following applies to all dogs: If they live in packs, then they can talk.
2. All huskies are dogs and live in packs.
3. Snowy is a husky.

Exercise 4 (Lists)
Implement the following Prolog predicates:

a) `last_but_one(+L, -E).` Return the last but one element of the list L in E.

b) `my_infix(+I, +L).` Test whether I is an infix of the list L.

c) `my_suffix(+S, +L).` Test whether S is a suffix of the list L.

d) `my_prefix(+P, +L).` Test whether P is a prefix of the list L.
For tasks b) to d), infix, suffix and prefix should be non-empty. Use append/3.

e) `del_element(+E, +L, -R).` Delete *all* occurences of E in the list L and return the resulting list in R.

Hint: Start with the base case (terminates recursion) when implementing recursive predicates.

Exercise 5 (Lists)
Implement a predicate `insert_at/4` which inserts an element to a list at a given index. If the index is larger than the size of the list, the element should be added to the end of the list. Examples:

```
1 insert_at(elm, [a,b,c], 2, [a,elm,b,c]).
2 insert_at(k, [a,b], 1, [k,a,b]).
3 insert_at(t, [a], 3, [a,t]).
```