DTU Course 02156 Logical Systems and Logic Programming

Mandatory Assignment 5 — Deadline Thursday 2/12 23:59

MUST BE SOLVED INDIVIDUALLY

You are only allowed to get help from the teacher and the teaching assistants.

You may use the notes and definitions on the first page of the sample exams.
You are allowed to use your computer and there is no 2 hours time limit.

For the whole assignment you must submit exactly 2 files on DTU Learn:

1. A single PDF file with extension `.pdf` with the report.
2. A single Prolog file with extension `.pl` with the programs.

Absolutely no ZIP files, no text processing documents or any other file formats.

The report should not contain any program listings (just refer to the Prolog file).

The programs should load in SWI-Prolog without any errors or warnings.
The programs must be properly documented with comments.
If possible use only the ISO Prolog features of SWI-Prolog covered in the course.

All programs must be tested and the tests must be included in the report.

In particular:

- Test a few normal cases.
- Test the special cases.
- Test with variables only (if the instantiation pattern allows variables).

Show the Prolog queries and the corresponding answers — and keep explanations short.

Problem 1 (20%)

In the following a semicolon (`;`) is used to separate the solutions to a query. This corresponds to the common use of the semicolon in an interactive Prolog session.

Question 1.1

Write a deterministic program `member1(+List)` that succeeds if and only if `List` is a non-empty list in which the first element occurs twice in the list. Sample queries:

```
?- member1([1,3,1,2]).

Yes

?- member1([1,3,4,2]).

No

?- member1([]).

No

?- member1([1]).

No

?- member1([1,1]).

Yes

?- member1([1,1,1]).

Yes
```

Only the following predicates can be used:

`member(?Elem,?List)`

`append(?List1,?List2,?List3)`

`sort(+List,?Sorted)`

`length(+List,?Integer)`

The problem continues on the next page

Question 1.2

Write a deterministic Prolog program `p(+Term,?List)` that succeeds if and only if `List` is a sorted list of all terms in `Term` that are not lists. A sample query:

```
?- member(L,[[a,[a,b,[],c],[[d]]],[123,[4,5]],[[[]]],f(a)]), p(L,R).

L = [a, [a, b, [], c], [[d]]]
R = [a, b, c, d] ;

L = [123, [4, 5]]
R = [4, 5, 123] ;

L = [[[]]]
R = [] ;

L = f(a)
R = [f(a)] ;

No
```

Only the following predicates can be used:

`member(?Elem,?List)`

`append(?List1,?List2,?List3)`

`sort(+List,?Sorted)`

`length(+List,?Integer)`

Problem 2 (20%)

Consider the following fragment of a word frequency list for a large English text:

```
w(5,2186369,a,det).
w(2107,4249,abandon,v).
w(5204,1110,abbey,n).
w(966,10468,ability,n).
w(321,30454,able,a).
w(6277,809,abnormal,a).
w(3862,1744,abolish,v).
w(5085,1154,abolition,n).
w(4341,1471,abortion,n).
w(179,52561,about,adv).
w(69,144554,about,prep).
w(3341,2139,above,a).
w(942,10719,above,adv).
w(786,12889,above,prep).
w(2236,3941,abroad,adv).
w(5106,1146,abruptly,adv).
```

The format is: `w(SortOrder,Frequency,Word,WordClass)`

`SortOrder` is `1` for the most frequent word. `WordClass` is the category: `det` for a determiner, `v` for verb, `n` for a noun, `a` for adjective, and so on.

You must test your programs on the complete list available in the file `database.pl` available as `boolean.txt` on DTU Learn — but do not include this file in your submission — instead start as follows in a new file:

```
:- ensure_loaded(database).
```

You are also encouraged to test your programs on the above fragment of the list shown above but do not include these tests in your submission.

Question 2.1

Write a Prolog program `selectlist/1` that returns a list of all words of category `n` (nouns) or category `v` (verbs) that are among the words with a sort order of 100 or less.

The list must not have duplicates and can be empty (as for the above fragment).

The order of the words in the list is not important.

The complete list has 20 such words.

The problem continues on the next page

Question 2.2

Write a Prolog program `dump` that finds all words that are both of category `a` (adjective) and category `adv` (adverbs), and for each such word prints a line for each additional category for the word (the line should consist of just the particular word and category), hence for the above fragment for example:

```
?- dump.
above prep

Yes
```

The complete list yields 28 extra lines.

Problem 3 (40%)

The details of the following answers must be provided and explained (in particular it is not sufficient to just list the result from a Prolog program).

Question 3.1

Consider the following formula: $\exists x \forall y p(x, y) \rightarrow \forall y \exists x p(x, y)$

Prove the formula using refutation, skolemization and the general resolution procedure.

Question 3.2

Consider the following formula: $(\forall x(p(x) \rightarrow \neg \exists y q(y, x))) \rightarrow (p(a) \rightarrow \neg q(a, a))$

Prove the formula using refutation, skolemization and the general resolution procedure.

Question 3.3

Consider the following formula: $(\forall x p(x) \wedge \forall x q(x)) \rightarrow \forall x(p(x) \wedge q(x))$

Prove the formula using refutation, skolemization and the general resolution procedure.

Question 3.4

Consider the following formula: $(\forall x \exists y(p(x) \wedge \neg p(y))) \rightarrow \neg q(a)$

Prove the formula using refutation, skolemization and the general resolution procedure.

Problem 4 (20%)

Use the online SeCaV Unshortener in order to prove the formula $\exists x \forall y p(x, y) \rightarrow \forall y \exists x p(x, y)$.

Insert the SeCaV Unshortener lines in the Prolog file as a comment.

Do not include the proof in Isabelle.

Sample lines for the formula $p \rightarrow p$:

```
/*

Imp p p

AlphaImp
  Neg p
  p
Ext
  p
  Neg p
Basic

*/
```