02156 Exercises-02

Jørgen Villadsen

2021

Special Exercise

The exercises are normally formulated such that they can function as exam problems and they should use only ISO Prolog features. However this initial exercise is special since it is a step-by-step introduction to certain features of the SWI-Prolog system.

Compile the following tiny program where `list(?List)` succeeds if and only if `List` is a list:

```
list(List) :- length(List,_).
```

The predicate `length(?List,?Integer)` is built-in and fails if `List` is not a list. Try the following three queries to see the compiled clauses:

```
?- listing.
?- listing(list).
?- listing(list/1).
```

The first query gives a listing of the clauses for all predicates, the second query only for the `list` predicate and the third query only for the `list` predicate with one argument. In this case the listing is the same for the three queries (except possibly for some system predicates to be ignored). Note that the variables are renamed in the listing.

Test it with at least the following queries:

```
?- list([]).
?- list(atom).
?- list(12345).
?- list([1,2,3,4,5]).
?- list([[a,b,c],12345,[x,y,z]]).
```

The predicate `length` can even be used to create a list holding only variables. This property also holds for the predicate `list` as can be seen from the following queries:

```
?- length(L,5).
?- length(L,N).
?- list(L).
?- list([1,2,3|L]).
```

For each query ask for more solutions if possible by entering `;` (semicolon) — or just use the spacebar. The variables in the lists created have special (unique) names starting with `_` (underscore).

For each of the following queries state the number of solutions and the answer substitutions:

```
?- member(X,[7,5,9,2,8,4,1,3,6]).
?- member(3,[7,5,X,2,8,Y,1,3,6]).
?- append([1,2,3],[a],Z).
?- append([1,2,3],Y,Z).
?- append(X,[a],Z).
?- append(a,a,[]).
?- append([],a,a).
?- member(a,L), member(b,L).
?- length(L,2), member(a,L), member(b,L).
?- length(L,3), member(a,L), member(b,L).
?- L=[_,_,_], member(a,L), member(b,L), append(_,[c,_],L).
?- length([X],X).
```

None of `length`, `member` and `append` are ISO Prolog predicates. In SWI-Prolog the predicate `length` is built-in, which means that it cannot be redefined, and the basic predicates `member` and `append` are library auto-loaded, which means that they can be redefined but are otherwise like built-in predicates. The SWI-Prolog definition of the `member` predicate is more efficient than the classical one but some queries will differ in the details. Thus it is sometimes better to just add the classical definition.

Exercise 1

Write a program `interval_member(?Integer,+List,+IntegerMin,+IntegerMax)` that succeeds if and only if `Integer` is a member of `List` and strictly between `IntegerMin` and `IntegerMax`.

```
?- interval_member(X,[7,5,9,2,8,4,1,3,6],3,6).
```

Here the solutions (first argument) should be `X = 5` and `X = 4` since only these members of the list (second argument) are strictly between `3` (third argument) and `6` (fourth argument).

Other predicates like `member` can be used. The program is very short and there are several ways to do it.

Exercise 2

Write a program `list(?List)` that succeeds if and only if `List` is a list.

Other predicates like `member` and `append` must not be used (the predicate `length` must not be used either).

Exercise 3

Write a program `select(?Elem,?List1,?List2)` that succeeds if and only if `List2` is `List1` with `Elem` removed (it removes one occurrence of `Elem` in `List1` to give `List2` and the elements are selected in the order of appearance in the list).

Other predicates like `member` and `append` must not be used (but perhaps consider `member` for inspiration).

Exercise 4

Write a program `ordered(+List)` that succeeds if and only if the elements in `List` are ordered by the `=<` relation.

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

Yes

?- ordered([1,3,2]).

No
```

Exercise 5

Write programs `one`, `two`, `three`, and `infinity` that succeeds one, two, three, and infinitely many times.

Other predicates like `member` and `append` must not be used (in particular the predicates `true` and `repeat` must not be used).

Exercise 6

Write a program `sublist(?List1,?List2)` that succeeds if and only if `List1` is a sublist of `List2` (the empty list is a sublist of any list).

Hint: Other predicates like `append` can be used and a sublist of a list is a prefix of a suffix of the list (equivalently: a suffix of a prefix).