TECHNICAL UNIVERSITY OF DENMARK (DTU)

Written Sample-Exam-1, 2021 Page 1 of 4 pages

Course: Logical Systems and Logic Programming

Course number: 02156

Exam duration: 2 hours

Aids allowed: All written works of reference

Weighting: Stated for each problem

The following basic predicates can be used when writing Prolog programs:

```
member(H,[H|_]).
member(H,[_|T]) :- member(H,T).

append([],U,U).
append([H|T],U,[H|V]) :- append(T,U,V).
```

Here `member(?Elem,?List)` succeeds if and only if `Elem` can be unified with one of the members of `List` and `append(?List1,?List2,?List3)` succeeds if and only if `List3` unifies with the concatenation of `List1` and `List2`.

Standard predicates like `is`, `fail`, `write`, `nl` and `findall` can also be used.

In the following a Prolog program is said to be deterministic if and only if it does not succeed more than once.

Assume available a deterministic predicate `sort(+List,?Sorted)` that can be used to sort a list. Duplicates are merged as shown in the following example:

```
?- sort([3,1,4,1,2],S).

S = [1, 2, 3, 4]

Yes
```

Assume also available a predicate `length(+List,?Integer)` that can be used to calculate the number of elements in a list.

Problem 1 (30%)

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

State the remaining solutions to the following query:

```
?- append(_,L,[1,2,3]), member(X,L).

L = [1, 2, 3]
X = 1 ;

L = [1, 2, 3]
X = 2 ;

L = [1, 2, 3]
X = 3 ;

...
```

Question 1.2

Consider the following Prolog program:

```
p([],[]).
p([_,X|T],[X|U]) :- p(T,U).
```

State the solutions to the following query:

```
?- member(L,[[],[1],[1,2],[1,2,3],[1,2,3,4],[1,2,3,4,5]]), p(L,R).
```

Question 1.3

Write a Prolog program `same_length(+List1,+List2)` that succeeds if and only if the two lists have the same number of elements.

Write two variants: one using only the predicate `length` and another not using any other predicate except possibly `same_length` (hence it can be recursive).

The problem continues on the next page

Sample queries:

```
?- same_length([],[]).

Yes

?- same_length([],[a]).

No

?- same_length([a,b],[b,a]).

Yes

?- same_length([a,b],[a]).

No
```

Problem 2 (30%)

Consider the following formula: $\exists x p(a,x,b) \vee \forall x \neg p(a,x,b)$

Question 2.1

Use refutation and the systematic construction of a semantic tableau. State whether this shows that the formula is valid or not.

Question 2.2

Use refutation, skolemization and the general resolution procedure. State whether this shows that the formula is valid or not.

Problem 3 (40%)

Consider the following definition of a particular directed graph:

```
edge(1,2).
edge(2,3). edge(2,4).
edge(3,3). edge(3,4). edge(3,5).
edge(4,3).
```

For example there is an edge from node 2 to node 3 (and from node 2 to node 4).

Node 2 has two out-going edges and node 5 has one in-going edge.

Question 3.1

Write a deterministic Prolog program `count` that for each node prints the difference between the number of in-going and out-going edges (the difference is negative if there are more out-going than in-going edges):

```
?- count.
1 -1
2 -1
3 0
4 1
5 1

Yes
```

Node 1 has no in-going edges and one out-going edge, hence the difference -1 is printed.

Question 3.2

Write a Prolog program `test(+Start,+End,?List)` that succeeds if and only if `List` is a path from the node `Start` to the node `End` with no repeated nodes.

Hint: Add an auxiliary recursive predicate that uses a list of nodes already visited.

For example there are two paths from node 1 to node 5 as the following queries show:

```
?- test(1,5,[1,2,3,5]).

Yes

?- test(1,5,[1,2,4,3,5]).

Yes
```