First Exam
Logic Programming, LVA 703113

June 25, 2015

Name: **Student Number**:

The exam consists of 5 exercises with a total of 100 (+2) points. Please fill out your name and credentials *before* you start the exam.

1	2	3	4	5	Sum

0–49: **5**	50–59: **4**	60–74: **3**	75–89: **2**	90–102: **1**

1. Consider the following (naive) implementation of *reverse/2*, where the implementation of *append/3* is standard.

```
reverse([],[]).
reverse([X],[X]).
reverse([X|Xs],Zs) :-
        reverse(Xs,Ys),
        append(Ys,[X],Zs).
```

   a) Construct the SLD tree of the ground query *reverse*([1,2,3],[3,2]) (based on Prolog's selection function). (5 pts)

   b) A logic program is called *deterministic*, if the cardinality of the answer set is one. Decide whether or not the program above is deterministic and explain your answer. (5 pts)

   c) Provide an improved implementation of *reverse/2* making use of an accumulator. (5 pts)

   d) Proof or disprove: any ground query for reverse with accumulator requires a proof tree of size (asymptotically) linear in the first argument to *reverse*. (10 pts)

2. Consider the predicate *occurrences(Sub,Term,N)* that is true if N is the number of occurrences of subterm Sub in $Term$, where you can assume $Term$ is ground.

   a) Implement *occurrences/3* as a recursive program, where as system predicates only types should be used. (10 pts)

   b) Reimplement *occurrences/3*, but this time make use of system calls and higher-order constructs. (5 pts)

3. Consider the following grammars G_1 and G_2

$$G_1 := P \to [P] \mid PP \mid \varepsilon \qquad G_2 := B \to [B \mid]B \mid \varepsilon\,.$$

   a) Implement the predicate *paren/2* as a DCG that generates $\mathsf{L}(G_1)$ by *directly* encoding G_1. Is this clever? (Explain your answer). (10 pts)

   b) Implement the predicate *string/4* as a DCG, such that string(L,R) generates $\mathsf{L}(G_2)$ and L (R) returns the number of left (right) brackets of the generate string. (10 pts)

   c) Suppose x is generated by paren *and* string(L,R). What can one then deduce for L and R? (Explain your answer). (5 pts)

4. Consider the following Prolog program.

```
foo(X,Y) :- foo([X|Xs]\Xs,Y,[X]).

foo([]\[],Y,Visited) :- !, fail.
foo([A|Xs]\Ys,A,Visited).
foo([A|Xs]\Ys,B,Visited) :-
        setof1(N,edge(A,N),Ns),
```

```
        foo2(Ns, Visited, Visited1, Xs\Ys, Xs1),
        foo(Xs1, B, Visited1).

foo2([N|Ns], Visited, Visited1, Xs, Xs1) :-
        member(N, Visited),
        foo2(Ns, Visited, Visited1, Xs, Xs1).
foo2([N|Ns], Visited, Visited1, Xs\[N|Ys], Xs1) :-
        \+ member(N, Visited),
        foo2(Ns, [N|Visited], Visited1, Xs\Ys, Xs1).
foo2([], V, V, Xs, Xs).

setof1(Template, Goal, Set) :-
        setof(Template, Goal, Set).
setof1(Template, Goal, Set) :-
        \+ setof(Template, Goal, Set), !, Set = [].
```

a) Give a declarative reading of the program. (8 pts)

b) The meaning of the program changes if *setof1*/3 is replaced by the system predicate *setof*/3. Give an example of a goal that succeeds in the original program, but fails in the altered program. (7 pts)

5. Determine whether the following statements are true or false. Every correct answer is worth 2 points; wrong answers "earn" -1 points. (22 pts)

statement	**yes**	**no**
(Prolog) terms are build from logical variables, constants, and functions.	☐	☐
A type is an infinite set of terms.	☐	☐
A rule is a universally quantified logical formula of the form $A \leftarrow B_1, B_2, \ldots, B_n$, where A is a goal and for all $i = 1, \ldots, n$: B_i is a goal.	☐	☐
An SLD refutation is a finite SLD derivation ending in $\square$.	☐	☐
A proof tree is the same as an SLD tree	☐	☐
A logic program's meaning is the Herbrand model of the program.	☐	☐
The operators `#=`, `#\=`, and `#>` are standard arithmetic comparison operators.	☐	☐
Answer Set Programming (ASP) is a novel programming language that extends CLP, that is, all constraint logic programs can be formulated as ASPs.	☐	☐
Prolog programs are executed using SLD resolution, where leftmost and topdown selection is used.	☐	☐
A cut fixes all choices between the moment of matching a rule's head with the goal of the parent clause and the cut.	☐	☐
The predicate *setof(Template,Goal,Bag)* is similar to *bagof*. However it removes duplicates in the obtained multiset, which is also sorted.	☐	☐