First Exam
Logic Programming, LVA 703113

January 30, 2017

Name:

The exam consists of 6 exercises with a total of 100 points. Please fill out your name and credentials *before* you start the exam. Lecture notes, text books, or smart devices like IBM's Watson are neither allowed nor necessary.

1	2	3	4	5	6	Sum

0–49: **5**	50–59: **4**	60–74: **3**	75–89: **2**	90–100: **1**

1. Consider the predicate subterm(Sub,Term), which is true if *Sub* is a subterm of *Term*.
   a) Implement the predicate such that it works as intended on ground terms. (10 pts)
   b) Construct the Prolog search tree of the ground query subterm(f(a),g(f(a),b)) (based on Prolog's selection function). (5 pts)
   c) Does the predicate work as intended (without sideeffect) on arbitrary terms? Explain your answer. (5 pts)

2. Consider the predicate bubblesort(List,SortedList), implemented as follows.

```
bubblesort(Xs,Ys) :-
    length(Xs,N),
    bsort(N,Xs,Ys).

bsort(0,Xs,Xs).
bsort(N,[X|Xs],Ys) :-
    N > 0,
    N0 is N - 1,
    bubble(X,Xs,Ys0),
    bsort(N0,Ys0,Ys).

bubble(X,[],[X]).
bubble(Y,[X|Xs],[X|Ys]) :-
    X < Y,
    bubble(Y,Xs,Ys).
bubble(Y,[X|Xs],[Y|Ys]) :-
    X >= Y,
    bubble(X,Xs,Ys).
```

   The intended meaning of bubblesort is the set of ground facts, where *List* is a number list and *SortedList* consists of the elements of *List* in increasing order.
   a) Consider bubble(Y,Xs,Ys) and prove by induction on the height of the search tree, that for all *increasing* Xs, Ys is the result of insertion Y in Xs at the correct position. (7 pts)
   b) Observe that bubble is iteratively swapping elements X and Y if they do not obey the order. Based on this observation, prove the correctness of bubblesort with respect to its intended meaning. (8 pts)
   c) Provide an alternative implementation of bubblesort. You may assume the existence of a type predicate ordered(Xs), expressing an increasing number list. (10 pts)

3. Implement a predicate *duplicate*/3 that duplicates the elements of a list a given number of times. For example the query duplicate([a,b,c],2,Xs) should deliver the answer Xs = [a, a, b, b, c, c]. Use difference-lists in your implementation, where you can assume that \ separates difference lists. (10 pts)

4. Consider the following grammar for propositional formulas over the atoms p, q, and r:

$P \to$ p | q | r $\quad\quad$ $P \to$ ~P
$P \to (P$ & $P)$ $\quad\quad$ $P \to (P$ | $P)$
$P \to (P$ => $P)$

- Write a DCG that generates the languages by *directly* encoding the grammar and builds an expression tree for the formula parsed. (5 pts)
- Improve your implementation by taking into account the following precedence of connectives ~ > & > | > =>, so that brackets can be dropped. (10 pts)

5. Implement the predicate non_member(X,Xs), where X is not an element of Xs in such a way that X can only be instantiated by a number and Xs by a list of numbers. (10 pts)

```
:- non_member(1,[_,_,_]).
```

6. Determine whether the following statements are true or false. Every correct answer is worth 2 points, every wrong answer -1 points. (The worst that can happen is that you get zero points for this exercise.) (20 pts)

statement	**yes**	**no**
In logic programming, terms are built from logical variables, constants and functions.	☐	☐
A computation of a goal G from a program P is the verification of an inference $P \vdash G$.	☐	☐
A type is an arbitrary, but finite set of terms.	☐	☐
We call a type complete, if it is closed under instantiation.	☐	☐
Difference lists are effective if independently different sections of a list are built, which are then concatenated.	☐	☐
Consider the standard implementation of *append*/3. Then any call to *append* terminates iff the second argument is a complete list.	☐	☐
A Prolog clause is called *tail recursive* iff it has one recursive call and zero or more calls to system predicates that appear before the recursive call.	☐	☐
A cut fixes all choices between (and including) the moment of matching the rule's head with parent goal and the cut. If backtracking should reaches the cut, then the cut succeeds and the execution is continued with the clause after the clause containing the cut.	☐	☐
Like almost any other programming language, answer set programming is Turing complete.	☐	☐
The predicate *bagof*($Template$,$Goal$,Bag) unifies Bag with the first alternative of $Goal$ that meets $Template$.	☐	☐