Rheinische Friedrich-Wilhelms-Universität Bonn
Institut für Informatik III
Prof. Dr. A. B. Cremers

Advanced Logic Programming
Summer Semester 2012
Dr. Günter Kniesel

# Assignment 5

Due: Sunday, 27.05.2011, 23:59:59 via SVN

Please note that the lecture and the exercise groups on Wednesday, May 23 will not take place because of the 'dies academicus'.

The week after the dies academicus is entirely free of courses (Whitsun break), so the results of this assignment may be submitted one week later than usual, on May 27.

## Task 1. *Double negation* (2 Points)

Given the following program discuss and argue whether r/1 and s/1 are equivalent or not. Provide a simple definition of p/2 on which you can demonstrate your arguments.

**Tip**: Consider the possible invocation modes of r/1 and s/1.

```
r(X) :- p(a,X).

s(X) :- not(not(p(a,X)).
```

## Task 2. *Classification and Negation* (3 Points)

Assume we have a database of results of tennis games played by members of a club. The results are represented as facts for the predicate beat/2, meaning that the player mentioned in the first argument has beaten the player in the second argument:

```
beat( tom, jim ).    % tom has beaten jim
beat( ann, tom ).    % ann has beaten tom
beat( pat, jim ).    % pat has beaten jim
```

Your task is to define a predicate "category(Player,Category)" that classifies the players into three categories:

1) ***winner:*** A player who won all his or her games.
2) ***fighter:*** A player who won some games and lost some.
3) ***loser:*** A player who lost all his or her games.

For instance, "?- category(tom, fighter)." should succeed.

## Task 3. *Grouping consecutive list elements* (3 Points)

Write a predicate that groups consecutive repeated elements of a list into sublists. If a list contains non-consecutive repeated elements they should be placed in separate sublists. Example:

?- group([1,1,1,1,2,c,c,1,1,d,e,e,e,e],X).

X = [[1,1,1,1],[2],[c,c],[1,1],[d],[e,e,e,e]]

## Task 4. *Grouping consecutive list elements* (1 Points)

Modify your predicate from Task 3 so that it does not put an element into a sublist if there is no consequtive repeated element. Example:

?- group([1,1,1,1,2,c,c,1,1,d,e,e,e,e],X).

X = [[1,1,1,1],2,[c,c],[1,1],d,[e,e,e,e]]

Tip: In both cases (Task 3 and 4) the essential question is "How can your predicate remember whether it had seen the same element in the previous step?"

## Task 5. *Understanding term-based predicates* (6 Points)

Try to understand the following uncommented predicate definition:

```
p([],[]).
p([A],[A]).
p([A|B],[A,C|D]):- p(B,[C|D]), A < C.
p([A|B],[C|D])  :- p(B,[C|E]), not(A<C), p([A|E],D).
```

Your task is to find out what the predicate does.

a) (*2 points*) Describe how each clause and each sub goal contributes to its functionality.
b) (*1 point)* Improve the readability of the predicate by proper renaming of the predicate and its variables.
c) (*2 points*) Write a comprehensive documentation that describes its general intention, the invocation modes in which it can be used and the differences between the modes (modes that behave the same way should be described together). Make the description as concise as possible by using (if appropriate) the ? mode as a generalisation of + and –.
d) (*1 point*) Exemplify each mode by a sample query and its result. This is useful for documentation and as a basis for automated testing.

Tip for c and d): When describing the predicate, consider any "type information" that you can deduce from its arguments. That includes not only the modes but also any constraints (implicit assumptions in the code) about the values passed in each argument.