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

Advanced Logic Programming
Summer Semester 2011
Dr. Günter Kniesel

Assignment 9

Due: Sunday, 19.06.2011, 23:59:59 via SVN

Submit your solution for this assignment into your group's SVN as an **Eclipse Project** named "assignment09".
Programming exercises should be submitted as **Prolog files** in this project.
Theoretical exercises should be submitted as **PDF files** in this project.

Task 1. *Cuts* (2 Points)

a) Consider the following program

```
p(1).
p(2):-!.
p(3).
```

Write all answers to the following queries:

```
1  ?- p(X).
2  ?- p(X), p(Y).
3  ?- p(X), !, p(Y).
```

b) Explain for each case why the program gives that answer.

Tip: As a preparation for the oral exam at the end of the course do it with *pen and paper only*, not by running the program! Or even better: Try to explain it to a colleague. For oral exams, practicing talking is the best preparation!

Task 2. *Declarative and Procedural Meaning and Cuts* (3 Points)

Give the declarative and procedural meaning for the predicates p, q, r. Describe the cause for differences and similarities. Recall that the declarative meaning of cut is simply "true" (it always succeeds).

```
p:- a, b.
p:- c.

q:- a, !, b.
q:- c.

r:- c.
r:- a, !, b.
```

Task 3. *Sorting and Cuts* (6 Points)

Where are good places to use cuts to optimise the following sorting programs? What would be changed by the cuts?

Which version (a, b, c and each either with or without cuts) would you recommend to use? Explain the reason behind your recommendation. Implement your preferred version and give it a comprehensive description.

```
a) sort([],[]).
   sort([X],[X]).
   sort([H|T1],[H,S|T2]):- sort(T1,[S|T2]), H < S.
   sort([H|T1],[S|T3])  :- sort(T1,[S|T2]), not(H<S), sort([H|T2],T3).

b) sort1([X|Xs], Ys) :- sort1(Xs, Zs), insert(X, Zs, Ys).
   sort1([],[]).

   insert(X, [], [X]).
   insert(X, [Y|Ys], [Y|Zs]  ):- X > Y, insert(X, Ys, Zs).
   insert(X, [Y|Ys], [X,Y|Ys]):- X=<Y.

c) sort2([X|Xs], Ys):-
       partition(Xs, X, Littles, Bigs),
       sort2(Littles, Ls),
       sort2(Bigs, Bs),
       append(Ls, [X|Bs], Ys).
   sort2([],[]).

   partition([X|Xs], Y, [X|Ls], Bs):- X=<Y, partition(Xs, Y, Ls, Bs).
   partition([X|Xs], Y, Ls, [X|Bs]):-  X>Y, partition(Xs, Y, Ls, Bs).
   partition([], Y, [], []).
```

Task 4. *Sorting and Cuts* (2 Points)

Think about different modes of invocation and different “kinds” of input Lists that illustrate relevant test cases for the above programs.

Afterwards use these inputs to compare the runtimes of the given programs and of your versions with cuts, as described in Task 2 of assignment sheet 6.