Hints:

- Please solve these exercises in groups of two!
- The solutions must be handed in directly before (very latest: at the beginning of) the exercise course on Wednesday, November 24th, 2010, in lecture hall AH 3. Alternatively you can drop your solutions into a box which is located right next to Prof. Giesl’s office (until the exercise course starts).
- Please write the names and immatriculation numbers of all (two) students on your solution. Also please staple the individual sheets!
- Exercises or exercise parts marked with a star are voluntary challenge exercises with advanced difficulty. However, they do not contribute to the overall maximum number of points that you can obtain by this exercise sheet.
- As the topics related to the exercises on this sheet are relevant for all versions of the lecture, all exercises on this sheet are relevant for all students.

Exercise 1 (Lifting Lemma): (4 + 4* points)

The Lifting Lemma (Lemma 3.4.8) states that given two clauses $K_1$ and $K_2$ with ground instances $K'_1$ and $K'_2$ and propositional resolvent $R'$ of $K'_1$ and $K'_2$, we have a resolvent $R$ of $K_1$ and $K_2$ w.r.t. predicate logic such that $R'$ is a ground instance of $R$.

a) Please show that the other direction of the Lifting Lemma does not hold, i.e., show that there are two clauses $K_1$ and $K_2$, a resolvent $R$ of $K_1$ and $K_2$ w.r.t. predicate logic and a ground instance $R'$ of $R$ such that there exist no ground instances $K'_1$ and $K'_2$ of $K_1$ and $K_2$ such that $R'$ is a resolvent of $K'_1$ and $K'_2$ w.r.t. propositional logic. The following diagram illustrates the situation.

\[ K_1 \quad K_2 \]
\[ \quad \rightarrow \quad R \]
\[ \quad R' \]
\[ \quad \neg \]
\[ K_1' \quad K_2' \]
\[ \quad \rightarrow \quad R' \]

\[ K_1'' \quad K_2'' \]

Hint: Read the the text in exercise part b) to get an idea for a counterexample.

b)* Consider a restricted version of resolution in predicate logic where we demand that all literals which become equal by applying the mgu are removed from the clauses. More precisely, a clause $R$ is a strict resolvent of two clauses $K_1$ and $K_2$ iff the following three conditions are satisfied:

- There are variable renamings $\nu_1$ and $\nu_2$ such that $\mathcal{V}(\nu_1(K_1)) \cap \mathcal{V}(\nu_2(K_2)) = \emptyset$.
- There are literals $L_1, \ldots, L_m \in \nu_1(K_1)$ and $L'_1, \ldots, L'_n \in \nu_2(K_2)$ with $m, n \geq 1$ such that
  \[
  \{L_1, \ldots, L_m, L'_1, \ldots, L'_n\} \text{ is unifiable with some mgu } \sigma \text{ and } \sigma(\nu_1(K_1) \setminus \{L_1, \ldots, L_m\}) \cap \sigma(\{L'_1, \ldots, L'_n\}) = \emptyset.
  \]
- $R = \sigma(\nu_1(K_1) \setminus \{L_1, \ldots, L_m\}) \cup (\nu_2(K_2) \setminus \{L'_1, \ldots, L'_n\})$

Please show that the inverse direction of the Lifting Lemma (like in the previous exercise part) again does not hold for this variant of resolution.
Exercise 2 (Incomplete Resolution): (3 points)

A clause \( R \) is an **unrenamed resolvent** of two clauses \( K_1 \) and \( K_2 \) iff the following two conditions are satisfied:

- There are literals \( L_1, \ldots, L_m \in K_1 \) and \( L'_1, \ldots, L'_n \in K_2 \) with \( m, n \geq 1 \) such that \( \{ \overline{L_1}, \ldots, \overline{L_m}, L'_1, \ldots, L'_n \} \) is unifiable with some mgu \( \sigma \).
- \( R = \sigma((K_1 \setminus \{ L_1, \ldots, L_m \}) \cup (K_2 \setminus \{ L'_1, \ldots, L'_n \})) \)

Unrenamed resolution is, thus, defined like resolution in predicate logic, but without renaming the clauses first such that they do not have any variables in common.

Please show that unrenamed resolution is incomplete. To this end, give a clause set and derive the empty clause from it with full resolution in predicate logic, but show that you cannot derive the empty clause with unrenamed resolution.

Exercise 3 (Multi-Resolution): (1 + 1 = 2 points)

In this exercise we consider an extension of resolution in propositional logic, which we call **multi-resolution**.

Let \( K_1 \) and \( K_2 \) be clauses without variables. Then a clause \( R \) is a **multi-resolvent** of \( K_1 \) and \( K_2 \) iff for some \( n > 0 \) there are literals \( L_1, \ldots, L_n \) such that \( K_1 = K'_1 \cup \{ L_1, \ldots, L_n \} \), \( K_2 = K'_2 \cup \{ \overline{L_1}, \ldots, \overline{L_n} \} \), and \( R = K'_1 \cup K'_2 \). The following diagram illustrates a multi-resolution step:

\[
\begin{align*}
K'_1 \cup \{ L_1, \ldots, L_n \} \\
K'_2 \cup \{ \overline{L_1}, \ldots, \overline{L_n} \} \\
\end{align*}
\]

Please prove or disprove the following statements:

- **a)** Multi-resolution is **sound**, i.e., there is no satisfiable clause set \( \mathcal{K} \) without variables from which one can derive \( \square \) by multi-resolution.

- **b)** Multi-resolution is **complete**, i.e., from any unsatisfiable clause set \( \mathcal{K} \) without variables one can derive \( \square \) by multi-resolution.

Exercise 4 (Binary Resolution): (2 points)

Please prove or disprove that binary resolution is complete for clause sets without variables, i.e., if \( \mathcal{K} \) is an unsatisfiable clause set without variables, can one derive \( \square \) by binary resolution?

Exercise 5 (Linear Resolution): (2 points)

Consider the clauses

\[ \{ p, \neg r \}, \{ \neg q, r \}, \{ \neg q, \neg r \}, \{ \neg p, \neg r \}, \{ q, r \}, \{ \neg p, \neg q \} \]

with \( p, q, r \in \Delta_0 \).

Resolve the empty clause using linear resolution.
Exercise 6 (SLD Resolution):  

(1 + 1 + 1 = 3 points)

Consider the clauses

\{
p(f(a), f(f(a)))
\}\, \{p(f(X), f(Y)), \neg p(X, f(Y))\}, \{\neg p(f(Z), f(f(Z)))\}, \{\neg p(f(a), f(f(b)))\}

with \( p \in \Delta_2 \), \( f \in \Sigma_1 \), \( a, b \in \Sigma_0 \).

a) Try to derive the empty clause with SLD resolution for both negative clauses in this clause set. If you cannot derive the empty clause, continue the derivation as long as possible.

b) Express the above clause set as queries, facts, and rules of a logic program.

c) Explain the meaning of the last derived clauses in the derivations in a) when the clauses are regarded as queries, facts, and rules of a logic program.