Please checkmark exercises that you solved before **11.01.2018**. The details of the checkmarking process will be available on the course website from **11.12.2017** onwards. Be sure to tick only those exercises which you can solve and explain on the blackboard. Do not leave the exercise work for the very last moment. Start preparing solutions as early as possible!

Problem 1. Let P_1 be the following problem:

$$\begin{array}{rl} brother\_of(hans, bob). & brother\_of(hans, luis). \\ sister\_of(mary, bob). & sister\_of(mary, luis). \\ brother\_of(nick, robin). & sibling\_of(ann, hans). \end{array}$$

$$\begin{array}{rcl} sibling\_of(X, Y) & \leftarrow & brother\_of(X, Y). \\ sibling\_of(X, Y) & \leftarrow & sister\_of(X, Y). \\ has\_sister(Y) & \leftarrow & sister\_of(X, Y). \\ has\_only\_brothers(Y) & \leftarrow & brother\_of(X, Y), \text{not } has\_sister(Y). \\ relative\_of(X, Y) & \leftarrow & sibling\_of(X, Y). \\ relative\_of(X, Z) & \leftarrow & sibling\_of(X, Y), relative\_of(Y, Z). \end{array}$$

Define the *Herbrand Universe* $HU(P_1)$, the *Herbrand Base* $HB(P_1)$ and three *Herbrand Models* I for P_1. What is the least model of P_1? For the definition of $HB(P_1)$ it is not necessary to explicitly enumerate all atoms but the range of all atoms has to be clear.

Solution:

- Herbrand universe, i.e., the set of all constants appearing in the program, is given as follows: $HU(P_1) = \{bob, hans, luis, mary, nick, robin, ann\}$
- Herband base, i.e., a set of facts formed by grounding predicates of the program with its constants in all possible ways, for the above program contains the following facts:
  $brother\_of(hans, hans)$, $brother\_of(hans, mary)$, $brother\_of(hans, bob)$ .....,
  $brother\_of(mary, mary)$, $brother\_of(mary, ...)$, ...
  .....
  $sister\_of(bob, hans)$, ...,
  $relative\_of(nick, mary)$, ...
  $has\_only\_brothers(bob)$...... .....
- Three examples of the Herbrand models for the above program are given below:
  $M_1 = \{brother\_of(hans, bob), brother\_of(hans, luis), brother\_of(nick, robin),$
  $sister\_of(mary, bob), sister\_of(mary, luis), sibling\_of(hans, bob), sibling\_of(hans, luis),$
  $sibling\_of(mary, bob), sibling\_of(mary, luis), sibling\_of(nick, robin), sibling\_of(ann, hans),$
  $has\_sister(bob), has\_sister(luis), has\_only\_brothers(robin), relative\_of(hans, bob),$
  $relative\_of(hans, luis), relative\_of(mary, bob), relative\_of(mary, luis), relative\_of(nick, robin),$
  $relative\_of(ann, hans), relative\_of(ann, bob), relative\_of(ann, luis)\}$

  $M_2 = M_1 \cup \{has\_sister(mary)\}$

  $M_3 = M_2 \cup \{has\_sister(nick)\}$.

- Only M_1 is the minimal (least) model for the given program, i.e., $LM(P_1) = M_1(P_1)$.

Problem 2. Let P_2 be the following normal logic program, where c and d are constants and X, Y are variables.

$$\begin{aligned}
\mathit{selected}(X) &\leftarrow \mathit{available}(X), \text{not } \mathit{not_selected}(X).\\
\mathit{not_selected}(X) &\leftarrow \mathit{available}(X), \text{not } \mathit{selected}(X).\\
\mathit{choice_made} &\leftarrow \mathit{selected}(X).\\
&\leftarrow \text{not } \mathit{choice_made}.\\
\mathit{available}(c).&\\
\mathit{available}(d).&
\end{aligned}$$

- Compute the grounding $grnd(P_2)$ of the program P_2.
- Formally check by computing the *Gelfond-Lifschitz reduct* whether the interpretation $I = \{\mathit{available}(c),\ \mathit{available}(d),\ \mathit{selected}(c),\ \mathit{not\_selected}(d),\ \mathit{choice\_made}\}$ is a stable model of P_2.

Solution:

- The grounding $grnd(P_2)$ is computed by replacing each variables in the rules with constants, appearing in P_2, i.e., c and d:

$$\begin{aligned}
\mathit{selected}(c) &\leftarrow \mathit{available}(c), \text{not } \mathit{not_selected}(c).\\
\mathit{not_selected}(c) &\leftarrow \mathit{available}(c), \text{not } \mathit{selected}(c).\\
\mathit{choice_made} &\leftarrow \mathit{selected}(c).\\
\mathit{selected}(d) &\leftarrow \mathit{available}(d), \text{not } \mathit{not_selected}(d).\\
\mathit{not_selected}(d) &\leftarrow \mathit{available}(d), \text{not } \mathit{selected}(d).\\
\mathit{choice_made} &\leftarrow \mathit{selected}(d).\\
&\leftarrow \text{not } \mathit{choice_made}.\\
\mathit{available}(c). &\quad \mathit{available}(d).
\end{aligned}$$

- We construct the reduct $grnd(P_2)^I$ as follows:

  1. Remove rules with "not $p(\dots)$", s.t. $p(...) \in I$:

$$\begin{aligned}
\mathit{selected}(c) &\leftarrow \mathit{available}(c), \text{not } \mathit{not_selected}(c).\\
\mathit{choice_made} &\leftarrow \mathit{selected}(c).\\
\mathit{not_selected}(d) &\leftarrow \mathit{available}(d), \text{not } \mathit{selected}(d).\\
\mathit{choice_made} &\leftarrow \mathit{selected}(d).\\
\mathit{available}(c). &\quad \mathit{available}(d).
\end{aligned}$$

2. Remove atoms "not $p(\dots)$" where $p(\dots) \notin I$:

$$\begin{aligned}
selected(c) &\leftarrow available(c).\\
choice_made &\leftarrow selected(c).\\
not_selected(d) &\leftarrow available(d).\\
choice_made &\leftarrow selected(d).\\
available(c). &\quad available(d).
\end{aligned}$$

3. Check if $I = \{available(c),\ available(d),\ selected(c),\ not\_selected(d),\ choice\_made\}$ is $LM(P_2^I)$. Since, it indeed holds that $I = LM(P_2^I)$, we have that the interpretation I is a stable model of P_2.

Problem 3.

1. Is the intersection of two Herbrand models of a normal logic program P again a Herbrand model? If yes, prove it, otherwise provide a counterexample.
2. Is the union of two Herbrand models of a positive logic program P again a Herbrand model? Again, if yes, prove it, otherwise provide a counterexample.

Solution:

1. The intersection of two Herbrand models of a normal logic program is not necessarily a Herbrand model. We show this by providing the following counterexample:

$$P = \{a \leftarrow not\ b. \quad b \leftarrow not\ a.\}$$

The program P has the following Herbrand models: $M_1 = \{a\}, M_2 = \{b\}$. However, we have $M_1 \cap M_2 = \emptyset$ is not a Herbrand model for this program. Indeed, e.g., the body of the first rule is satisfied, since $b \notin I_1$, but the head is not, as $a \notin I_1$.

2. The union of two Herbrand models of a positive logic program P might not be a its Herbrand model. Consider the following counterexample:

$$P = \{a \leftarrow b, c.\}$$

Let $M_1 = \{b\}$ and $M_2 = \{c\}$. Both M_1 and M_2 satisfy P; however, $M_1 \cup M_2 \not\models P$, since $M_1 \cup M_2 \models b$, and $M_1 \cup M_2 \models c$, i.e., the body of the rule is satisfied, but the head is not, i.e, $M_1 \cup M_2 \not\models a$.

Problem 4. Let a, b be constant symbols and X, Y be variables. Consider

$$\begin{aligned}
q(a). &\quad r(b).\\
s(X, Y, Z) &\leftarrow q(X),\ r(Y),\ f(Z).\\
f(X) &\leftarrow q(X),\ not\ r(X).
\end{aligned}$$

(i) Compute the grounding $grnd(P)$ of P.

(ii) Decide using the *Gelfond-Lifschitz reduct* whether the interpretation $I = \{q(a), r(b), f(a), s(a,b,a)\}$ is a stable model of P.

Solution:

(i) The grounding of the program $grnd(P)$ comprises the following rules:

$$
\begin{aligned}
s(a,a,a) &\leftarrow q(a), r(a), f(a).\\
s(a,a,b) &\leftarrow q(a), r(a), f(b).\\
s(a,b,a) &\leftarrow q(a), r(b), f(a).\\
s(a,b,b) &\leftarrow q(a), r(b), f(b).\\
s(b,a,a) &\leftarrow q(b), r(a), f(a).\\
s(b,a,b) &\leftarrow q(b), r(a), f(b).\\
s(b,b,a) &\leftarrow q(b), r(b), f(a).\\
s(b,b,b) &\leftarrow q(b), r(b), f(b).\\
f(a) &\leftarrow q(a), \mathit{not}\ r(a).\\
f(b) &\leftarrow q(b), \mathit{not}\ r(b).\\
q(a). &\quad\ \ r(b).
\end{aligned}
$$

(ii) The reduct $grnd(P)^I$ is as follows:

$$
\begin{aligned}
s(a,a,a) &\leftarrow q(a), r(a), f(a).\\
s(a,a,b) &\leftarrow q(a), r(a), f(b).\\
s(a,b,a) &\leftarrow q(a), r(b), f(a).\\
s(a,b,b) &\leftarrow q(a), r(b), f(b).\\
s(b,a,a) &\leftarrow q(b), r(a), f(a).\\
s(b,a,b) &\leftarrow q(b), r(a), f(b).\\
s(b,b,a) &\leftarrow q(b), r(b), f(a).\\
s(b,b,b) &\leftarrow q(b), r(b), f(b).\\
f(a) &\leftarrow q(a).\\
q(a). &\quad\ \ r(b).
\end{aligned}
$$

I is a model of the facts $q(a)$ and $r(b)$. Accordingly, $I \models f(a) \leftarrow q(a)$. Furthermore, since $q(a)$, $r(b)$, and $f(a)$ are in I, $s(a,b,a)$ has to be in I in order for I to satisfy the rule $s(a,b,a) \leftarrow q(a),\ r(b),\ f(a)$. This indeed holds, and moreover, I satisfies all of the remaining rules, since none of their bodies is in I. Therefore, we have that I is a model of $grnd(P)^I$. Moreover, it is easy to verify that I is also minimal, and hence it is a stable model of P.

Problem 5. Define a *normal logic program* P consisting of the ground atoms q, r, f and s that exhibits the following properties.

- P has exactly 3 answer sets,
- $\{q, s\}$ is an answer set of P, and

- P contains no more than 1 fact.

Solution:
Consider the program P containing the following rules:

$$
\begin{aligned}
&q.\\
&s \leftarrow q, \text{not } f, \text{not } r.\\
&f \leftarrow q, \text{not } s, \text{not } r.\\
&r \leftarrow q, \text{not } s, \text{not } f.
\end{aligned}
$$

The answer sets of P are $\{s, q\}$, $\{f, q\}$, and $\{r, q\}$

Problem 6. Consider a small fragment of the program for solving the *Project-Assignment Problem* (i.e., assigning managers to various industrial projects)

- Managers are represented by facts of the form
  $manager(aaron)$. $manager(bill)$. $manager(charlotte)$. etc.
- Projects are represented by facts of the form
  $project(p1)$. $project(p2)$. $project(p3)$. etc.
- Assignments of projects to managers are represented by facts of the form
  $assigned(P, M)$, meaning that the project P is assigned to the manager M.
- The managers can express their competence for leading a specific project. The preferences range from 0 ("I am not competent in the subject area of the project") to 3 ("I am really competent and want to lead the project"). The preferences are defined by facts of the form $pref(M, P, B)$ with the meaning that the manager M assessed his competence for the project P with B, where B is a constant from $\{0, 1, 2, 3\}$ with the following meanings:

  0: "I am not competent in the subject area of the project",

  1: "I can lead the project, but not particularly willing to",

  2: "I am willing to lead this project",

  3: "I really want to lead this project".

Your tasks are the following.

- Use the syntax of `DLV` and aggregate atoms to define a predicate $count(M, C)$ that counts for a specific manager M all projects that are assigned to M and stores this value in C.
- Use the above defined predicate $count(M, C)$ to specify the following constraints:

  (i) *Projects assigned only to managers who rated their competence in the respected subject area with 0 should be completely excluded.*

  (ii) *For any manager who got at least one project assigned to him/her, the sum of his/her preferences for the assigned projects should be greater or equal than twice the number of projects assigned to that manager.*

  Again use the syntax of `DLV` and aggregate atoms for the definition of this constraint.

Solution:

- % Count assigned projects per manager
  `count(M, C):-manager(M),#count{P:assigned(P,M)}.`
- Use `count` predicate to specify the required constraints:
  (i) exclude projects assigned only to managers who rated their competence for the project with 0

  % variant 1: ignore projects, for which at least one of the assigned managers did not provide his rating

  % if a project `P` assigned to a manager `M` was rated, infer `rated(M,P)`
  `rated(M,P):-assigned(P,M),pref(M,P,B).`

  % collect projects that were not rated by at least 1 assigned manager
  `not_all_rated(P):-assigned(P,M), not rated(M,P).`

  % among projects that were rated by all of the managers assigned to them, exclude those, which were assigned only to non-competent managers
  `:-project(P),not not_all_rated(P), #sum{B:pref(M,P,B),assigned(P,M)}=0.`

  % variant 2: (less precise) exclude projects that are only assigned to non-competent managers or to managers who did not specify their competence for the given project
  `:-project(P),#sum{B,P:pref(M,P,B),assigned(P,M)}=0`

  (ii) % for managers with at least 1 assigned project, the sum of preferences for the assigned projects should be $\geq 2*$number of assigned projects

```
:-manager(M), count(M,C), C>=1,
  #sum{B:pref(M,P,B),assigned(P,M)}=N, N<2*C.
```

Problem 7. Imagine a transport network represented using facts $link(c, d)$, where c, d denote bus stops and $link(c, d)$ states that there is a direct bus connection from c to d. Define a normal logic program that uses a predicate $not\_accessible(c, d)$ to calculate all vertices d that are not accessible from c. (A node d is not accessible from a node c if there is no direct connection from c to d, and if there is no bus route from c to d in the network. A bus route in a network is a sequence of bus stops such that from each of these bus stops there is a direct connection to the next bus stop in the sequence.)

Solution: We construct the following program:

$$
\begin{aligned}
node(X) &\leftarrow link(X, Y).\\
node(Y) &\leftarrow link(X, Y).\\
\\
route(X, Y) &\leftarrow link(X, Y).\\
route(X, Y) &\leftarrow route(X, Z), link(Z, Y).\\
\\
not\_accessible(X, Y) &\leftarrow node(X), node(Y), not\ route(X, Y).
\end{aligned}
$$

Problem 8. For a program P, we denote by $AS(P)$ the set of all answer sets of P. Let P, Q be programs. We say that P, Q are

- equivalent, if $AS(P) = AS(Q)$ and
- strongly equivalent if $AS(P \cup R) = AS(Q \cup R)$ for every program R.

Prove or refute that if

1. whenever (ii) holds then also (i) and
2. the converse holds, i.e., whether (i) implies (ii).

Solution: Obviously, (ii) implies (i) (just take $R = \emptyset$). However, (i) does not imply (ii): take $P = \{a \vee b\}$, $Q = \{a \leftarrow \mathit{not}\ b.\quad b \leftarrow \mathit{not}\ a\}$, and $R = \{a \leftarrow b.\quad b \leftarrow a.\}$. Then $Q \cup R$ has no answer set, while $P \cup R$ has the answer set $\{a, b\}$. However, P and Q are equivalent, since $AS(P) = AS(Q) = \{\{a\}, \{b\}\}$.

Problem 9. Assume that an electrical station has two entrance points represented by the constants *nothern_entrance* and *southern_entrance*. An entrance is accessible, if it is not known to be closed. If an entrance is closed, then it is definitely not accessible. Use the predicates $\mathit{accessible}(X)$ (entrance X is accessible) and $\mathit{closed}(X)$ (entrance X is closed) and define a disjunctive logic program that has exactly the following answer sets:
$A_1 = \mathit{closed}(\mathit{nothern\_entrance}), \mathit{accessible}(\mathit{southern\_entrance}), \neg \mathit{accessible}(\mathit{nothern\_entrance})$
$A_2 = \mathit{closed}(\mathit{southern\_entrance}), \mathit{accessible}(\mathit{nothern\_entrance}), \neg \mathit{accessible}(\mathit{southern\_entrance})$

Solution: Both models represent that one of the entrances is open while the other is closed.

$$\begin{aligned}
\neg \mathit{accessible}(X) &\leftarrow \mathit{closed}(X). \\
\mathit{accessible}(X) &\leftarrow \text{not } \mathit{closed}(X). \\
\mathit{closed}(\mathit{northern_entrance}) &\vee \mathit{closed}(\mathit{southern_entrance}). \\
\mathit{accessible}(\mathit{northern_entrance}) &\vee \mathit{accessible}(\mathit{southern_entrance}).
\end{aligned}$$

Problem 10. Bob, Alice and Jerry went hiking and they cannot be reached by phone. You are their friend and want to figure out where they could be. You have the following information:

- Every person is either at the lake or in the forest but not both.
- Jerry cannot be at the lake without Alice.
- Alice is for sure not with Bob at the lake.
- At least one person is in the forest.

Define a logical program P and compute its answer sets. Use $\mathit{forest}(X)$ and $\mathit{lake}(X)$ as predicate symbols and *bob*, *alice* and *jerry* as constant symbols.

Solution:

$$
\begin{aligned}
&\text{\% Every person is either at the lake or at the forest}\\
&lake(X) \ \vee \ forest(X).\\
&\text{\% ...but not both}\\
&\leftarrow lake(X), forest(X).
\end{aligned}
$$

$$
\begin{aligned}
&\text{\% Jerry cannot be at the lake without Alice}\\
&\leftarrow lake(jerry), not\ lake(alice).
\end{aligned}
$$

$$
\begin{aligned}
&\text{\% Alice is for sure not with Bob at the lake}\\
&\leftarrow lake(alice), lake(bob).
\end{aligned}
$$

$$
\begin{aligned}
&\text{\% At least one person is in the forest}\\
&ok \leftarrow forest(X).\\
&\leftarrow not\ ok.
\end{aligned}
$$

The answer sets of the above programs are as follows:

$I1 = \{forest(jerry), forest(bob), lake(alice), ok\}$
$I2 = \{lake(jerry), forest(bob), lake(alice), ok\}$
$I3 = \{forest(jerry), lake(bob), forest(alice), ok\}$
$I4 = \{forest(jerry), forest(bob), forest(alice), ok\}$