Logic Programming SS17
Exercise Sheet 6 (due Friday, June 30th, 2017)

Prof. Dr. Jürgen Giesl | Jera Hensel

Notes:

- Please solve these exercises in **groups of three or four**!
- The solutions must be handed in **directly before (very latest: at the beginning of)** the exercise course on Friday, June 30th, 2017, in lecture hall **AH 2**. Alternatively you can drop your solutions into a box which is located right next to Prof. Giesl's office (this box is emptied **15 minutes before** the exercise course starts).
- Please write the **names** and **immatriculation numbers** of all students on your solution. Also, please staple the individual sheets!

Exercise 1 (μ-recursion): (2 + 2 + 1 + 1 + 3 + 2 = 11 points)

Please show that the following five functions `odd`, `ite`, `max`, $\mathtt{f}_d$, and `mod` are μ-recursive by expressing each function using only the six principles from Def. 4.2.1. Which of these functions are primitive recursive?

Hints:

- To show that a function is primitive recursive, express it using only the first **five** principles.
- To show that a function is not primitive recursive, give a reasonable explanation.
- You may use the predefined μ-recursive functions `plus`, `times`, `minus`, and `div` and their implementation as a logic program from the lecture.

a) $\mathtt{odd}(x) = \begin{cases} 1, & \text{if } x \text{ is odd} \\ 0, & \text{otherwise} \end{cases}$ for all $x \in \mathbb{N}$

Hint: You may use the primitive recursive function `minus` from the lecture.

b) $\mathtt{ite}(x, y, z) = \begin{cases} y, & \text{if } x > 0 \\ z, & \text{otherwise} \end{cases}$ for all $x, y, z \in \mathbb{N}$

c) $\mathtt{max}(x, y) = \begin{cases} x, & \text{if } x > y \\ y, & \text{otherwise} \end{cases}$ for all $x, y \in \mathbb{N}$

d) Consider the following two functions $\mathtt{f}_d$ and $\mathtt{g}_d$ with $\mathtt{f}_d \colon \mathbb{N}^2 \to \mathbb{N}$ and $\mathtt{g}_d \colon \mathbb{N}^3 \to \mathbb{N}$.
$\mathtt{f}_d(x, y) = z$ iff $\mathtt{g}_d(x, y, z) = 0$ and for all $0 \leq k < z$, $\mathtt{g}_d(x, y, k)$ is defined and $\mathtt{g}_b(x, y, k) > 0$ with
$\mathtt{g}_d(x, y, z) = \begin{cases} x \cdot y - z, & \text{if } x \cdot y \geq z \\ 0, & \text{otherwise} \end{cases}$

e) For all $x, y \in \mathbb{N}$, we have $\mathtt{mod}(x, y) = \begin{cases} \mathrm{mod}(x, y), & \text{if } x \neq 0 \neq y \text{ for } x, y \in \mathbb{N} \\ 0, & \text{if } x = 0, y \in \mathbb{N} \\ \text{undefined}, & \text{otherwise} \end{cases}$

Here, $\mathrm{mod}(x, y)$ is the modulo operation that computes the remainder after division of x by y.

f) Following the proof for Theorem 4.2.5, write a logic program $\mathcal{P}$ with a predicate $\mathtt{mod}(X, Y, Z)$ such that for all $x, y, z \in \mathbb{N}$, we have $\mathtt{mod}(x, y) = z$ iff $\mathcal{P} \models \mathtt{mod}(x, y, z)$ (cf. exercise part **e**)).

Exercise 2 (SLD tree): (6 points)

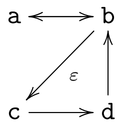
Consider the following logic program $\mathcal{P}$:

```
path(X, X, Y).
path(X, Y, s(Z)) :- edge(X, A), path(A, Y, Z).
path(X, Y, Z) :- eps(X, A), path(A, Y, Z).

edge(a, b).
edge(b, a).
edge(c, d).
edge(d, b).

eps(b, c).
```

The predicates `edge` and `eps` define the following graph $\mathcal{G}$:



Furthermore, `path(X, Y, Z)` is true iff there is a path from `X` to `Y` in $\mathcal{G}$ where at most `Z` non-ε-edges are used along the path. As an example, `?- path(a, X, s(0))` gives the solutions `X = a`, `X = b`, and `X = c`. Here, natural numbers are represented by the function symbols `0` and `s` (i.e. `s(0)` stands for 1, `s(s(0))` stands for 2, etc.)
Please give the SLD tree for the query `?- path(c, c, s(s(0)))`. Subtrees that Prolog explores after having found the second solution should be abbreviated with dots (...).

Exercise 3 (SLD tree): (6 + 1 = 7 points)

Consider the following logic program $\mathcal{P}$:

```
p(X,X,s(Y)) :- q(s(X),Y), p(X,Y,a).
p(0,s(0),a).
q(0,0).
q(s(X),s(Y)) :- q(X,Y).
```

a) Give the resulting SLD tree for the query `p(X, Y, Z)` up to depth four. Here, the query is at depth zero. Use ∞ to indicate infinite paths and $\lightning$ to mark nodes that cannot be evaluated further. Also give all answer substitutions. Which solutions does Prolog find? Explain your answer.

b) Change the order of exactly two literals in the clauses of $\mathcal{P}$ such that for the query above the resulting SLD tree is finite.