Rule-based Programming

Assignment #1

Exercise 1 (SICStus Prolog)**.** Make yourself familiar with *SICStus Prolog* (which is installed in the Linux lab.)

- Information on SICStus Prolog can be found on `www.sics.se/sicstus/`. Read the chapter "How to Run Prolog".
- Install Emacs support for SICStus: Add the line
  `(load "/opt/sicstus4.0.1/lib/sicstus-4.0.1/emacs/sicstus_emacs_init")`
  to the file `~/.emacs` (where `~` is your home directory).
- Write a "Hello world!" program in SICStus Prolog:

```
:- use_module(library(chr)).
:- chr_constraint hello/0.

hello :- write('Hello world!').
```

  Compile the program in Emacs and run it with the query: `hello.`

Exercise 2. Implement the following programs which consist of *one* rule each. (Remember to insert an adequate program header!)

- `p1 @ p <=> q.`
- `p2 @ p ==> q.`
- `p3 @ p,q <=> true.`
- `p4 @ p \ q <=> true.`

For each program, pose the following queries:

(1) `p`
(2) `q`
(3) `p,p`
(4) `q,q`

Explain the different answers of the system.

Exercise 3. Implement the following programs which consist of *one* rule each.

- `p1 @ p(a) <=> true | true.`
- `p2 @ p(X) <=> X=a | true.`
- `p3 @ p(X) <=> true | X=a.`
- `p4 @ p(X) <=> true , X = a | true.`
- `p5 @ p(X) <=> X = a , X = b | true.`

For each program, pose the following queries:

(1) `p(a)`
(2) `p(b)`
(3) `p(X)`

Explain the different answers of the system.

Exercise 4. Implement the following programs which consist of *one* rule each.

- `p1 @ p(X,Y), q(Z,Y) <=> q(X,Y).`
- `p2 @ q(Z,Y), p(X,Y) <=> q(X,Y).`
- `p3 @ p(X,Y), q(Z,Y) ==> q(X,Y).`
- `p4 @ p(X,Z)\ q(Z,Y) <=> q(X,Y).`

For each program, pose the following queries:

(1) `p(a,b), q(b,c)`
(2) `p(A,B), q(B,C)`
(3) `p(A,B), q(B,C), p(D,A)`
(4) `p(Y,C), q(C,A), q(C,A)`

Explain the different answers of the system.