AIT Semantic and Declarative Technologies Course

Class practice

1. Splitting atoms
   The prefix of an atom A is an atom P, if P consists of the first few characters of the atom A, in the same order.

```
% atom_prefix(+Atom, ?Prefix, +N): Prefix is a prefix of Atom, and has length N.
% In other words: the forst N characters of Atom forms the atom Prefix.

| ?- atom_prefix(abcde, Prefix, 0).        ---->   Prefix = '' ? ; no
| ?- atom_prefix(abcde, Prefix, 3).        ---->   Prefix = abc ? ; no
| ?- atom_prefix(abcde, Prefix, 5).        ---->   Prefix = abcde ? ; no
| ?- atom_prefix(abcde, Prefix, 6).        ---->   no
```

   Hint: Use atom_codes/2, length/2 and append/3.

2. Accumulating some terms in a compound.

```
% sum_nums(+C, ?Sum): Sum is the sum of all integers in the compound C.
| ?- sum_nums(a, S).                              ---->   S = 0 ? ; no
| ?- sum_nums(1, S).                              ---->   S = 1 ? ; no
| ?- sum_nums(f(X,[1,3,b],g(2,1,a0)), S).         ---->   S = 7 ? ; no
```

3. Substituting a value:
   Write a Prolog predicate that finds the substitution value of a constant based on a substitution list. The elements of the substitution list are in the form of Name-Value pairs, vhere Name is an atom, and Value is the substitution value of that atom. The substitution value of any number is itself, regardless of the substitution list. If an atom is not in the substitution list, its value is 0. If an atom appears multiple times in the substitution list, the first appearance must be used.

```
% substitution(+C, +SL, ?V): The value of the constant (atom or number) C based on the
% substitution list SL is V
| ?- substitution(y, [x-1,y-2,z-3], V).       ---->   V = 2 ? ; no
| ?- substitution(u, [x-1,y-2,z-3], V).       ---->   V = 0 ? ; no
| ?- substitution(x, [x-1,z-3,x-2], V).       ---->   V = 1 ? ; no
| ?- substitution(4, [x-1,y-2,z-3], V).       ---->   V = 4 ? ; no
```

4. Calculate the value of an expression:
   Using substitution/3, write a Prolog predicare to calculate the evaluation of an expression with multiple variables, based on a substitution list. The expression is given with a Prolog datastructure consisting of atoms and integers combined with one and two argument operators that can be evaluated with the built in predicate 'is'.

```
% evaluation(+Expr, +SL, ?Value): the expression Expr can be evaluated using the
% substituion list SL to Value
| ?- evaluation(x, [x-22], V).                      ---->   V = 22 ? ; no
| ?- evaluation(-x, [x-5], V).                      ---->   V = -5 ? ; no
| ?- evaluation(33, [x-22], V).                     ---->   V = 33 ? ; no
| ?- evaluation((x+y)*(x+y)+1, [x-1,y-2], V).       ---->   V = 10 ? ; no
| ?- evaluation(x*abs(z)+x, [x-1,z-(-2)], V).       ---->   V = 3 ? ; no
```