

## Prolog – Arithmetic

- Prolog is a programming language, therefore, arithmetic is implemented as expected.
- The only difference to other programming languages is that assignment is done via the predicate `is` rather than the equal sign, since the equal sign has been used for the unification operator.

### Examples:

```
?- X is 10 + 5;  
X = 15
```

```
?- X is 10 + 5 * 6 / 3;  
X = 20
```

← Precedence and associativity of operators are respected.

## Prolog – Arithmetic

Example: write a predicate definition for `length/2` that takes a list in its first argument and returns the length of the list in its second argument.

```
length([], 0).  
length(L, N) :- L = [H|T], length(T, NT), N is NT + 1.
```

## More Arithmetic...

- (1) Define a predicate `max/3` that takes two numbers as its first two arguments and unifies the last argument with the maximum of the two.
- (2) Define a predicate `maxlist/2` that takes a list of numbers as its first argument and unifies the second argument with the maximum number in the list. The predicate should fail if the list is empty.
- (3) Define a predicate `ordered/1` that takes a list of numbers as its argument and succeeds if and only if the list is in non-decreasing order.

## Prolog – Arithmetic

Example: we can also use arithmetic in compound statements.

```
?- X is 5, Y is 2 * X.  
X = 5  
Y = 10
```

## Prolog – I/O

- **write(term)**
  - is true if term is a Prolog term, writes term to the terminal.
- **read(X)**
  - is true if the user types a term followed by a period, X becomes unified to the term.
- **nl**
  - is always true and writes a newline character on the terminal.

☞ Extra-logical predicates due to the side-effect of writing/reading to/from the terminal.

## Prolog – I/O

```
?- write(tom).  
tom  
  
?- write([1,2]).  
[1, 2]  
  
Prolog I/O Prompt → ?- read(X).  
|: boo.  
X = boo  
  
?- read(Q).  
|: [1,2,3].  
Q = [1, 2, 3]
```

## Prolog – I/O

Example: write a predicate definition for fadd/1 that takes a list of integers, adds 1 to each integer in the list, and prints each integer onto the terminal screen.

```
fadd([ ]).  
fadd([ H | T ]) :- H is H + 1, write(H), nl, fadd(T).
```