Priority of operators

$Author: mkraska $
$Date: 2018-09-06 12:30:28 +0200 (Do, 06 Sep 2018) $

Operators, Basics
Mathematics
Contents

If an operand stands between two operators, then the result depends on which of the operators takes higher priority.

Expressions with operators of equal priority

Expressions with multiple operators of equal precedence are evaluated strictly from left to right. This means that the leftmost operator uses it's adjacent operands and the result forms an operand in the remaining expression.

$a+b+c+d$ is equivalent to $((a+b)+c)+d$

You also may imagine the expression being transformed to function form using a two-arguments function plus(2):

$plus(plus(plus(a,b),c),d)$ with $plus(a,b):=a+b$

In the operator form, the sub-expression b+c is syntactically correct, whereas in the function form there is no syntactically correct sub-expression which just inlcudes operands b and c.

The nasty implication is that you have no chance to highlight b+c e.g. for bracketing or for replacing it by something else. Particularly beginners can be extremly frustrated by this.

If you want to evaluate b+c before adding a and d, you must use brackets.

$a+(b+c)+d$ $plus(plus(a,plus(b,c)),d)$

Expressions with operators of varying priority

Besides using brackets, there is another way to control sequence of evaluation: operator priority. If an operand stands between two operators, then it is used by the higher priority operator or by the left one if both are of equal priority.

$(a+b)+c$ This bracket is not needed because the left-to-right rule applies for operators of equal priority

$a+b\cdot c$ b becomes operand of the operator "times", because times has higher priority than plus.

Writing readable expressions with as few brackets as possible requires well designed priority levels, close to how they are understood in ordinary mathematics.
SMath is not always successful in that respect.

```
Level 1 (highest):    - brackets
                      - functions
                      - array structures (sys, mat, line)
                      - operators that internally are functions
                      - multiplicative result split (for unit setting)

Level 2               - unary faculty operator

Level 3               - multiply, divide

Level 4               - add, subtract (binary operators)
                      - unary minus, plus/minus, minus/plus
                      - assignment

Level 5               - boolean operators

Level 6               - numeric evaluation
                      - symbolic evaluation
```