# Generating Software from Specifications WS 2013/14 - Solution 5

## Solution for Exercise 13

The directory `blatt5/calendarChkSol` contains a file `CalendarCheckSol.fw` which specifies a calendar processor, that solves all subtasks of this exercise. The solutions are explained and inserted into the prepared FunnelWeb macros. Overridden variants are deactivated.

Be aware that the abstract syntax has been influenced by a symbol mapping and a rule mapping.

1. Day numbers in the year are computed using the function dayInYear, which is im plemented in the provided C module.

   Two of the three concrete productions for `Date` are mapped on to one abstract RULE. The third can not be mapped, too, because `Month` is a terminal.

2. The attribute `minDate` has to be computed in every context with `Date` on the right-hand side, and from there up to the root. In alternatives of `DateDescr` which do not have a `Date` in their subtree, `minDate` is set to a value bigger than any day in a year (i.e. bigger than 366).

3. Using a CONSTITUENTS-WITH construct simplifies the value propagation drastically.

4. The minimum value is propagated down from the root to all occurrences of a `Date`. There it is compared to `Date.dayNum` and printed if it is minimal.

5. Using an INCLUDING construct simplifies the propagation.

6. Only the computations of the numbers of days in a year have to remain in two rules; all others are turned into symbol computations.

## Solution for Exercise 14

The directory `blatt5/calendarChkSol` contains a specification file `StatementsSol.fw` which specifies a little statement language. The solutions of the subtasks are inserted into the prepared FunnelWeb macros. The solutions of some subtasks are deactivated as required in the assignment. Symbol computations are used wherever possible.

## Solution for Exercise 15

No solution is given here.

## Solution for Exercise 16

No solution is given here.