


# Chapter 0

## Exercises

Course “Compiler Construction”

Martin Steffen

Spring 2022



# Section

## Exercises 05 (ST)

1. Postfix string as attribute
2. Simple Pascal-style type declarations
3. Dependency graphs and evaluation
4. Attribute grammar for classes

Chapter 0 “Exercises”
Course “Compiler Construction”
Martin Steffen
Spring 2022






# Exercises 05 (ST)

1. Postfix string as attribute
2. Simple Pascal-style type declarations
3. Dependency graphs and evaluation
4. Attribute grammar for classes

Exercises 05 (ST)
1. Postfix string as attribute
2. Simple Pascal-style type declarations
3. Dependency graphs and evaluation
4. Attribute grammar for classes

# 5.1: Original grammar

$$
\begin{array}{rcl}
exp & \rightarrow & exp + term \mid exp - term \mid term \\
term & \rightarrow & term * factor \mid factor \\
factor & \rightarrow & ( exp ) \mid \mathbf{number}
\end{array}
$$



# 5.1: Original AG for evaluation

| | productions/grammar rules | | | semantic rules |
|---|---|---|---|---|
| 1 | $exp_1$ | $\rightarrow$ | $exp_2 + term$ | $exp_1$ .val $= exp_2$ .val $+ term$ .val |
| 2 | $exp_1$ | $\rightarrow$ | $exp_2 - term$ | $exp_1$ .val $= exp_2$ .val $- term$ .val |
| 3 | $exp$ | $\rightarrow$ | $term$ | $exp$ .val $= term$ .val |
| 4 | $term_1$ | $\rightarrow$ | $term_2 * factor$ | $term_1$ .val $= term_2$ .val $* factor$ .val |
| 5 | $term$ | $\rightarrow$ | $factor$ | $term$ .val $= factor$ .val |
| 6 | $factor$ | $\rightarrow$ | ( $exp$ ) | $factor$ .val $= exp$ .val |
| 7 | $factor$ | $\rightarrow$ | **number** | $factor$ .val $=$ **number**.val |

# 5.2: (Type declarations): Underlying grammar

$$
\begin{aligned}
\mathit{decl} &\rightarrow \mathit{var\text{-}list} : \mathit{type} \\
\mathit{var\text{-}list} &\rightarrow \mathit{var\text{-}list} \ \mathbf{,\ id} \mid \mathbf{id} \\
\mathit{type} &\rightarrow \mathbf{integer} \mid \mathbf{real}
\end{aligned}
$$



# 5.3 (evaluation): Artificial AG

| productions/grammar rules | | | semantic rules |
|---|---|---|---|
| $S$ | $\rightarrow$ | $ABC$ | $B.\mathrm{u} = S.\mathrm{u}$ |
| | | | $A.\mathrm{u} = B.\mathrm{v} + C.\mathrm{v}$ |
| | | | $S.\mathrm{v} = A.\mathrm{v}$ |
| $A$ | $\rightarrow$ | $a$ | $A.\mathrm{v} = 2 * A.\mathrm{u}$ |
| $B$ | $\rightarrow$ | $b$ | $B.\mathrm{v} = B.\mathrm{u}$ |
| $C$ | $\rightarrow$ | $c$ | $C.\mathrm{v} = 1$ |



# 5.3: Changed AG

| production/grammar rule | | | semantic rules |
|---|---|---|---|
| $S$ | $\rightarrow$ | $ABC$ | $B.\mathrm{u} = S.\mathrm{u}$ |
| | | | $C.\mathrm{u} = A.\mathrm{v}$ |
| | | | $A.\mathrm{u} = B.\mathrm{v} + C.\mathrm{v}$ |
| | | | $S.\mathrm{v} = A.\mathrm{v}$ |
| $A$ | $\rightarrow$ | $a$ | $A.\mathrm{v} = 2 * A.\mathrm{u}$ |
| $B$ | $\rightarrow$ | $b$ | $B.\mathrm{v} = B.\mathrm{u}$ |
| $C$ | $\rightarrow$ | $c$ | $C.\mathrm{v} = C.\mathrm{u} - 2$ |



# 5.4: AG for classes

| productions/grammar rules | | | semantic rules |
|---|---|---|---|
| $class$ | $\rightarrow$ | **class name** $superclass$ **{** $decls$ **}** | |
| $decls$ | $\rightarrow$ | $decls$ **;** $decl$ | |
| $decls$ | $\rightarrow$ | $decl$ | |
| $decl$ | $\rightarrow$ | $variable\text{-}decl$ | not to be filled out |
| $decl$ | $\rightarrow$ | $method\text{-}decl$ | |
| $method\text{-}decl$ | $\rightarrow$ | $type$ **name** **(** $params$ **)** $body$ | |
| $type$ | $\rightarrow$ | **int** | |
| $type$ | $\rightarrow$ | **bool** | |
| $type$ | $\rightarrow$ | **void** | |
| $(superclass$ | $\rightarrow$ | **name**$)$ | filled by lexer |