

Chapter 0

Exercises

Course “Compiler Construction”

Martin Steffen

Spring 2022



Section

Exercises 06

1. Collateral vs. sequential declarations
2. Expression evaluation
3. Type conversion/evaluation (floats and int)
4. Type equality and type checking
5: “Ambiguous” syntax

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

6.1: Underlying grammar

$$
\begin{array}{rcl}
S & \rightarrow & exp \\
exp & \rightarrow & (\ exp\ ) \mid exp + exp \mid \mathbf{id} \mid num \mid \mathbf{let}\ dec\text{-}list\ \mathbf{in}\ exp \\
dec\text{-}list & \rightarrow & dec\text{-}list,\ decl \mid decl \\
decl & \rightarrow & \mathbf{id} = exp
\end{array}
$$



Exercises 06

1. Collateral vs. sequential declarations
2. Expression evaluation
3. Type conversion/evaluation (floats and int)
4. Type equality and type checking
5: "Ambiguous" syntax

6:1 Attributes (original)

remember: symbol tables & `intab` and `outtabl`.

symbol	attributes	kind
exp	`symtab`	inherited
	`nestlevel`	inherited
	err	synthesized
dec-list, *decl*	`intab`	inherited
	`outtab`	synthesized
	`nestlevel`	inherited
id	`name`	injected by scanner



Exercises 06

1. Collateral vs. sequential declarations
2. Expression evaluation
3. Type conversion/evaluation (floats and int)
4. Type equality and type checking
5: "Ambiguous" syntax

6:1 Sequential decl's (1)

Grammar Rule	Semantic Rules
$S \rightarrow exp$	$exp.symtab = emptytable$ $exp.nestlevel = 0$ $S.err = exp.err$
$exp_1 \rightarrow exp_2$ **+** exp_3	$exp_2.symtab = exp_1.symtab$ $exp_3.symtab = exp_1.symtab$ $exp_2.nestlevel = exp_1.nestlevel$ $exp_3.nestlevel = exp_1.nestlevel$ $exp_1.err = exp_2.err$ **or** $exp_3.err$
$exp_1 \rightarrow$ **(** exp_2 **)**	$exp_2.symtab = exp_1.symtab$ $exp_2.nestlevel = exp_1.nestlevel$ $exp_1.err = exp_2.err$
$exp \rightarrow$ **id**	$exp.err =$ **not** $isin(exp.symtab,$**id**$.name)$
$exp \rightarrow$ **num**	$exp.err =$ **false**
$exp_1 \rightarrow$ **let** $dec\text{-}list$ **in** exp_2	$dec\text{-}list.intab = exp_1.symtab$ $dec\text{-}list.nestlevel = exp_1.nestlevel + 1$ $exp_2.symtab = dec\text{-}list.outtab$ $exp_2.nestlevel = dec\text{-}list.nestlevel$ $exp_1.err = (decl\text{-}list.outtab = errtab)$ **or** $exp_2.err$

2

3

6.1: Sequential decl's (2)

$dec\text{-}list_1 \rightarrow dec\text{-}list_2$, $decl$	$dec\text{-}list_2.intab = dec\text{-}list_1.intab$ $dec\text{-}list_2.nestlevel = dec\text{-}list_1.nestlevel$ $decl.intab = dec\text{-}list_2.outtab$ $decl.nestlevel = dec\text{-}list_2.nestlevel$ $dec\text{-}list_1.outtab = decl.outtab$	4
$dec\text{-}list \rightarrow decl$	$decl.intab = dec\text{-}list.intab$ $decl.nestlevel = dec\text{-}list.nestlevel$ $dec\text{-}list.outtab = decl.outtab$	4
$decl \rightarrow$ **id** = exp	$exp.symtab = decl.intab$ $exp.nestlevel = decl.nestlevel$ $decl.outtab =$ **if** ($decl.intab = errtab$) **or** $exp.err$ **then** $errtab$ **else if** ($lookup(decl.intab,$**id**$.name) =$ $decl.nestlevel$) **then** $errtab$ **else** $insert(decl.intab,$**id**$.name, decl.nestlevel)$	1



Exercises 06

1. Collateral vs. sequential declarations
2. Expression evaluation
3. Type conversion/evaluation (floats and int)
4. Type equality and type checking
5: "Ambiguous" syntax



Exercises 06

1. Collateral vs. sequential declarations

2. Expression evaluation

3. Type conversion/evaluation (floats and int)

4. Type equality and type checking

5: “Ambiguous” syntax

6.2: expr. evaluation



Original expression grammar (again):

$$
\begin{array}{rcl}
S & \rightarrow & exp \\
exp & \rightarrow & (\ exp\ ) \mid exp + exp \mid \mathbf{id} \mid num \mid \mathbf{let}\ dec\text{-}list\ \mathbf{in}\ exp \\
dec\text{-}list & \rightarrow & dec\text{-}list,\ decl \mid decl \\
decl & \rightarrow & \mathbf{id} = exp
\end{array}
$$

now assume *sequential* declarations

Exercises 06

1. Collateral vs. sequential declarations
2. Expression evaluation
3. Type conversion/evaluation (floats and int)
4. Type equality and type checking
5. "Ambiguous" syntax



Exercises 06

1. Collateral vs. sequential declarations
2. Expression evaluation
3. Type conversion/evaluation (floats and int)
4. Type equality and type checking
5: “Ambiguous” syntax

6.3: conversion/evaluation

$$\begin{array}{rcl} exp & \rightarrow & exp + exp \mid exp - exp \mid exp * exp \mid exp \,/\, exp \\ & \mid & (\, exp \,) \mid \mathbf{num} \mid \mathbf{num} \,.\, \mathbf{num} \end{array}$$



Exercises 06

1. Collateral vs. sequential declarations
2. Expression evaluation
3. Type conversion/evaluation (floats and int)
4. Type equality and type checking
5: "Ambiguous" syntax



Exercises 06

1. Collateral vs. sequential declarations
2. Expression evaluation
3. Type conversion/evaluation (floats and int)
4. Type equality and type checking
5: “Ambiguous” syntax

6.4: type checking and type equality

Original grammar:

$$
\begin{array}{rcl}
\mathit{program} & \rightarrow & \mathit{var\text{-}decls}\ \mathbf{;}\ \mathit{fun\text{-}decls}\ \mathbf{;}\ \mathit{stmts} \\
\mathit{var\text{-}decls} & \rightarrow & \mathit{var\text{-}decls}\ \mathbf{;}\ \mathit{var\text{-}decl} \mid \mathit{var\text{-}decl} \\
\mathit{var\text{-}decl} & \rightarrow & \mathbf{id}\ \mathbf{:}\ \mathit{type\text{-}exp} \\
\mathit{type\text{-}exp} & \rightarrow & \mathbf{int} \mid \mathbf{bool} \mid \mathbf{array}\ [\ \mathit{num}\ ]\ \mathbf{:}\ \mathit{type\text{-}exp} \\
\mathit{fun\text{-}decls} & \rightarrow & \mathbf{fun}\ \mathbf{id}\ (\ \mathit{var\text{-}decls}\ )\ \mathbf{:}\ \mathit{type\text{-}exp}\ \mathbf{;}\ \mathit{body} \\
\mathit{body} & \rightarrow & \mathit{exp} \\
\mathit{stmts} & \rightarrow & \mathit{stmts}\ \mathbf{;}\ \mathit{stmt} \mid \mathit{stmt} \\
\mathit{stmt} & \rightarrow & \mathbf{if}\ \ \mathit{exp}\ \ \mathbf{then}\ \ \mathit{stmt} \mid \mathbf{id}\ \mathbf{:=}\ \mathit{exp} \\
\mathit{exp} & \rightarrow & \mathit{exp} + \mathit{exp} \mid \mathit{exp}\ \mathbf{or}\ \mathit{exp} \mid \mathit{exp}\ [\ \mathit{exp}\ ] \mid \mathbf{id}\ (\ \mathit{exps}\ ) \\
 & \mid & \mathbf{num} \mid \mathbf{true} \mid \mathbf{false} \mid \mathbf{id} \\
\mathit{exps} & \rightarrow & \mathit{exps}\ \mathbf{,}\ \mathit{exp} \mid \mathit{exp}
\end{array}
$$



Exercises 06

1. Collateral vs. sequential declarations
2. Expression evaluation
3. Type conversion/evaluation (floats and int)
4. Type equality and type checking
5: "Ambiguous" syntax



Exercises 06

1. Collateral vs. sequential declarations
2. Expression evaluation
3. Type conversion/evaluation (floats and int)
4. Type equality and type checking
5: "Ambiguous" syntax

Ambiguous C syntax

```
1    (A) - x
```





Exercises 06

1. Collateral vs. sequential declarations
2. Expression evaluation
3. Type conversion/evaluation (floats and int)
4. Type equality and type checking
5: "Ambiguous" syntax