Seminar 2

You are expected to present solutions to these problems at the seminar in week 2.

1 Suppose that there is a context-free grammar for program statements without a terminating semicolon with start symbol *statement*. Construct an EBNF grammar for a sequence of

- **a.** zero or more statements with a semicolon after each statement.
- **b.** one or more statements with a semicolon between statements.
- **c.** zero or more statements with a semicolon between statements.

2 Construct grammars on canonical form for the languages in the previous problem.

3 The following grammar generates a language on the alphabet $\{(,)\}$.

$$\begin{aligned} S &\rightarrow \texttt{"("}\ S\text{'}\texttt{)'} \\ S &\rightarrow SS \\ S &\rightarrow \texttt{"("}\ \texttt{')'} \end{aligned}$$

- **a.** Which strings with length 6 belong to the language?
- **b.** The grammar is ambiguous. Which is the shortest string in the language with at least two parse trees?

4 The following grammar is ambiguous. Construct an unambiguous grammar accepting the same language.

$$\begin{aligned} S &\rightarrow \texttt{"("}\ S\text{'}\texttt{)'} \\ S &\rightarrow SS \\ S &\rightarrow \epsilon \end{aligned}$$

5 The following grammar for logical expressions is ambiguous.

$$\begin{aligned} E &\rightarrow \texttt{"!"}\ E \\ E &\rightarrow E\text{'}\texttt{\&\&}\text{'}E \\ E &\rightarrow E\text{'}\texttt{——}\text{'}E \\ E &\rightarrow \text{ID} \end{aligned}$$

Assume that '!' has higher precedence than '&&' which in turn precedes over '——' . Construct an unambiguous grammar that respects the precedences describing the same language.

6 Construct a canonical grammar that is equivalent to the following EBNF rule.

$$\mathit{callStmt} \rightarrow \text{ID}\text{'}(\text{'}\ (\ \epsilon \mid \mathit{expr}\ (\text{','}\mathit{expr}\ )^*\ )\ \text{')'}$$

7 Every language that is described by a regular expression is generated by a *regular grammar*, i.e. a grammar where all productions have one of the forms

$$\begin{aligned} A &\rightarrow aB \\ A &\rightarrow a \\ A &\rightarrow \epsilon \end{aligned}$$

where A and B are non-terminals and a is a terminal. Construct such a grammar for the language described by the regular expression $(a^*b)|(ba^*)$. Is it possible to do it without any production of the form $A \rightarrow a$?

8 Construct an EBNF grammar that generates the language of all (basic) regular expressions over the alphabet $\{a, b\}$ respecting operator precedences. Notice that each string in this language is a regular expression. Some short strings in this language are: $\emptyset$, a, b, a^*, $a \cdot b$, $(a \mid b)^*$.