Concurrence et Compilation 2001

(Exercises)

1 Overview and Foundations

Keywords 1

Give short explanations of the following keywords:

Compiler, *interpreter*, *scanner*, *parser*, *language*, *grammar*.

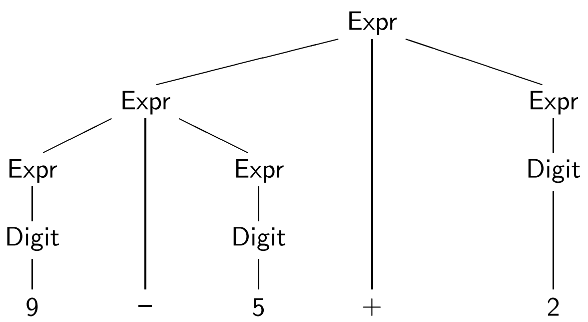
In particular, what are the similarities and differences between a compiler and an interpreter?

Exercise 1

The following *grammar* produces expressions consisting of digits $\mathsf{0},\ldots,\mathsf{9}$ separated by $+$ and $-$ signs.

$$
\begin{array}{lcl}
\mathsf{Expr} & ::= & \mathsf{Expr}\ \text{"}+\text{"}\ \mathsf{Expr} \\
 & | & \mathsf{Expr}\ \text{"}-\text{"}\ \mathsf{Expr} \\
 & | & \mathsf{Digit} \\
\mathsf{Digit} & ::= & \text{"0"} \mid \text{"1"} \mid \text{"2"} \mid \text{"3"} \mid \text{"4"} \mid \text{"5"} \mid \text{"6"} \mid \text{"7"} \mid \text{"8"} \mid \text{"9"}
\end{array}
$$

As an example, consider the expression $\mathsf{9} - \mathsf{5} + \mathsf{2}$. It can be derived by the following *concrete syntax tree*:



(a) Give a concrete syntax tree deriving $\mathsf{3} + \mathsf{7} - \mathsf{4} - \mathsf{2}$.

(b) Can you derive $- \mathsf{5} + \mathsf{2}$ from the grammar? If this is the case, give a concrete syntax tree; otherwise, explain why this is so.

A grammar is called *ambiguous* if it can have more than one concrete syntax tree generating a given string of *tokens*, that is, a given expression in the case of this exercise.

(c) Is the given grammar ambiguous? If this is so, try to find an unambiguous grammar producing the same *language*, that is, the same set of expressions.