Exercise VII

Finishing the MINI compiler

The parser of the MINI compiler has to be extended with proper error recovery. Furthermore this exercise is an opportunity to clean up and document your code, before your final handin.

An LL(1) parser like the parser for MINI signals an error in its input when it is confronted with a token in the input that is not one of the expected ones. Error recovery then has to skip tokens until the lexer recognizes a token with which the parser can resume its task. For this purpose the algorithm

```
ACTION skip tokens (>SYMBOL_SET)
```

is provided in `lexer.k3`, which will read tokens until the current token is in `SYMBOL_SET` or is an eof symbol. Such a set of tokens is often called a *synchronization set*.

When the current parser encounters a syntactic error, it will report this error correctly, skip to the end of file and then complain several times about missing constructs and tokens while climbing out of recursion. Clearly this form of error recovery is unwanted for a production compiler.

Unfortunately there is no such thing as a definite algorithm for the construction of synchronization sets (although a lot of literature exists on this subject). Typically one would take the follower set of a nonterminal as a good synchronization set. However, by using the context in which the syntactic error occurs, one may actually improve the error recovery behaviour of the parser.