CS 352, Jens Palsberg Sep 13, 2000

Homework 2: LL(1) parsing

Consider the grammar

$$
\begin{array}{rcl}
e & ::= & \texttt{true} \\
 & | & \texttt{false} \\
 & | & \texttt{!}\ e \\
 & | & e\ o\ e \\
o & ::= & \texttt{\&} \\
 & | & \texttt{|} \\
 & | & \texttt{\&\&} \\
 & | & \texttt{||}
\end{array}
$$

where $\{e, o\}$ is the set of non-terminal symbols, e is the start symbol, and $\{$ `true`, `false`, `!`, `&`, `|`, `&&`, `||` $\}$ is the set of terminal symbols. The grammar generates a subset of the Java expressions of type boolean. The various boolean operators have precedences as in Java (see the Java specification).

Rewrite the grammar into a grammar which is LL(1), and use the rewritten grammar as the basis for implementing a recursive descent parser in Java. If the input can be parsed correctly, then the parser should output the expression in postfix notation.

Submit on paper the LL(1) grammar, the FIRST and FOLLOW sets for each nonterminal symbol, and the predictive parsing table. Argue that the grammar is LL(1).

Submit electronically your program. Your main file should be called `Parse.java`, and if `Exp` is a file, possibly containing a Java boolean expression according to the above grammar, then

```
java Parse < Exp
```

outputs

- **either** the postfix version of the input and "Expression parsed successfully",
- **or** "Parse error in line ⟨linenumber⟩".