March 6, 2014

Jani Vihervä̈s
jani.vihervas@cs.helsinki.fi

Mini-PL Interpreter

58144 Compilers Project

Grammar:

$$
\begin{aligned}
< prog > &\rightarrow < stmts > \\
< stmts > &\rightarrow < stmt > \ ; \ < stmts' > \\
< stmts' > &\rightarrow \epsilon \mid < stmts > \\
< stmt > &\rightarrow \textbf{var} \ < ident' > \ : \ < type >< stmt' > \\
&\mid \ < ident > \ := \ < expr > \\
&\mid \textbf{for} \ < ident > \ \textbf{in} \ < expr > \ .. \ < expr > \ \textbf{do} \ < stmts > \ \textbf{end for} \\
&\mid \textbf{read} \ < ident > \\
&\mid \textbf{print} \ < expr > \\
&\mid \textbf{assert} \ (< expr > \ ) \\
< stmt' > &\rightarrow \epsilon \mid := \ < expr > \\
< expr > &\rightarrow \ < opnd >< op >< opnd > \\
< expr' > &\rightarrow \epsilon \mid < unary > \\
< opnd > &\rightarrow \ < int > \\
&\mid \ < string > \\
&\mid \ < expr' >< bool > \\
&\mid \ < ident > \\
&\mid \ (\ < expr > \ ) \\
< type > &\rightarrow \textbf{int} \mid \textbf{string} \mid \textbf{bool} \\
< reserved \ \ keyword > &\rightarrow \textbf{var} \mid \textbf{for} \mid \textbf{end} \mid \textbf{in} \mid \textbf{do} \mid \textbf{read} \mid \textbf{print} \mid \textbf{int} \mid \textbf{string} \mid \textbf{bool} \mid \textbf{assert} \\
< unary > &\rightarrow \textbf{!} \\
< op > &\rightarrow + \mid \textbf{-} \mid \textbf{*} \mid / \mid < \mid > \mid <= \mid >= \mid = \mid \textbf{\&}
\end{aligned}
$$

$< ident' >$ adds identifier to symbol table, where as $< ident >$ looks the identifier from the symbol table. Operators >, <= and >= are added, because they are very easy to implement.

Predict sets:

Production	Predict set
$< prog >$	**var**, $< ident >$, **for**, **read**, **print**, **assert**
$< stmts >$	**var**, $< ident >$, **for**, **read**, **print**, **assert**
$< stmts' >$	$$ **var**, $< ident >$, **for**, **read**, **print**, **assert**
$< stmt >$	**var** $< ident >$ **for** **read** **print** **assert**
$< stmt' >$	; :=
$< expr >$	`i`, `s`, **!**, `b`, $< ident >$, (
$< expr' >$	`b` **!**

$< opnd >$	`i`
	`s`
	!, `b`
	$< ident >$
	(

$< type >$	**int** **string** **bool**