March 6, 2014

Jani Viherväs
jani.vihervas@cs.helsinki.fi

Mini-PL Interpreter

58144 Compilers Project

Grammar:

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

$< ident' >$ 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**