

March 6, 2014

Jani Viherväs  
jani.vihervas@cs.helsinki.fi

# MINI-PL INTERPRETER

58144 COMPILERS PROJECT

Grammar:

```
< prog > → < stmts >
< stmts > → < stmt > ; < stmts' >
< stmts' > → ε | < stmts >
< stmt > → var < ident' > : < type > < stmt' >
           | < ident > := < expr >
           | for < ident > in < expr > .. < expr > do < stmts > end for
           | read < ident >
           | print < expr >
           | assert ( < expr > )
< stmt' > → ε | := < expr >
< expr > → < opnd > < op > < opnd >
< expr' > → ε | < unary >
< opnd > → < int >
           | < string >
           | < expr' > < bool >
           | < ident >
           | ( < expr > )
< type > → int | string | bool
< reserved keyword > → var | for | end | in | do | read | print | assert
                      | int | string | bool | true | false
< unary > → !
< op > → + | - | * | / | < | > | <= | >= | = | &
```

< 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
$\langle prog \rangle$	var, $\langle ident \rangle$ , for, read, print, assert
$\langle stmts \rangle$	var, $\langle ident \rangle$ , for, read, print, assert
$\langle stmts' \rangle$	\$\$ var, $\langle ident \rangle$ , for, read, print, assert
$\langle stmt \rangle$	var $\langle ident \rangle$ for read print assert
$\langle stmt' \rangle$	; :=
$\langle expr \rangle$	i, s, !, b, $\langle ident \rangle$ , (
$\langle expr' \rangle$	b !
$\langle opnd \rangle$	i s !, b $\langle ident \rangle$ (
$\langle type \rangle$	int string bool