CP471 Assignment 2

Aaron Exley

February 14, 2020

LL(1) Language

<program> ::= <fdecls> <declarations> <statement_seq>.
<fdecls> ::= <fdec>; <fdecls> | ϵ
<fdec> ::= def <type> <fname> (<params>) <declarations> <statement_seq> fed
<params> ::= <type> <var><params_rest> |
<params_rest> ::= , <params> | ϵ
<fname> ::= <id>

<declarations> ::= <decl>; <delarations> | ϵ
<decl> ::= <type> <varlist>
<type> ::= int | double
<varlist> ::= <var><varlist_rest>
<varlist_rest> ::= , <varlist> | ϵ

<statement_seq> ::= <statement><statement_seq_rest>
<statement_seq_rest> ::= ; <statement_seq> | ϵ

<statement> ::= <var> = <expr> |
if <bexpr> then <statement_seq> <if_rest> |
while <bexpr> do <statement_seq> od |
print <expr> |
return <expr> |
<if_rest> ::= fi | else <statement_seq> fi |

<expr> ::= <term><expr_rest>
<expr_rest> ::= + <term><expr_rest> | - <term><expr_rest> | ϵ
<term> ::= <factor><term_rest>
<term_rest> ::= * <factor><term_rest> | / <factor><term_rest> | % <factor><term_rest> | ϵ
<factor> ::= <var> | <number> | (<expr>) | <fname>(<exprseq>)
<exprseq> ::= <expr><exprseq_rest>
<exprseq_rest> ::= , <exprseq> | ϵ

<bexpr> ::= <bterm><bexpr_rest>
<bexpr_rest> ::= or <bterm><bexpr_rest> | ϵ
<bterm> ::= <bfactor><bterm_rest>
<bterm_rest> ::= and <bfactor><bterm_rest> | ϵ
<bfactor> ::= (<bfactor_rest>) | not <bfactor>
<bfactor_rest> ::= <bexpr> | <expr> <comp> <expr>

<comp> ::= < | > | == | <= | >= | <>

<var> ::= <id> | <id>[<expr>]

<letter> ::= a | b | c | ... | z
<digit> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0
<id> ::= <letter> | <id><letter> | <id><digit>

<number> ::= <integer> | <double>

First

First(<program>) = {def, }
First(<fdecls>) = {def, ϵ}
First(<fdec>) = {def}
First(<params>) = {int, double,}
First(<params_rest) = {, , ϵ }
First(<fname>) = {id}

First(<declarations>) = First(<decl>) = First(<type>) = {int, double}

First(<varlist>) = {id}
First(<varlist_rest>) = {, , ϵ}

First(<statement_seq>) = First(<statement>) = {id, if, while, print, return}
First(<statement_seq_rest>) = {; , ϵ}

First(<if_rest>) = {fi, else}

First(<expr>) = First(<term>) = First(<factor>) = First(<exprseq>) = {id, integer, double, (}
First(<expr_rest>) = {+, -, ϵ}
First(<term_rest>) = {*, /, %, ϵ}
First(<exprseq_rest>) = {, , ϵ}

First(<bexpr>) = First(<bterm>) = First(<bfactor>) = {(, not}
First(<bexpr_rest>) = {(or, ϵ}
First(<bterm_rest>) = {(and, ϵ}
First(<bfactor_rest>) = {id, integer, double, (, not}

First(<comp>) = {<, >, ==, <=, >=, <>}

First(<var>) = {id}

Follow