ebnf_rules.ebnf

Author: Callum France

```
<ident> := ('a' | 'b' | 'c' | 'd' | 'e' | 'f' | 'g' | 'h' | 'i' | 'j' | 'k' | 'l' |
'm' | 'n' | 'o' | 'p' | 'q' | 'r' | 's' | 't' | 'u' | 'v' | 'w' | 'x' | 'y' | 'z')+

<number> := ('0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9')+

<id_num> := <ident> | <number>

<term> := <id_num>, {('*' | '/'), <id_num>}

<expression> := <term>, {('+' | '-'), <term>}

<compound_statement> := 'START', {<statement>, ';'}, <statement>, 'STOP'

<for_statement> := 'FOR', <ident>, ':=', <expression>, 'DO', {<statement>, ';'},
<statement>, 'ROF'

<do_statement> := 'DO', {<statement, ';'}, <statement>, 'WHILE', <expression>, 'OD'

<while_statement> := 'WHILE', <expression>, 'DO', {<statement, ';'}, <statement>,
'ELIHW'

<if_statement> := 'IF', <expression>, 'THEN', <statement>, 'FI'

<procedure_call> := 'EXECUTE', <ident>

<assignment> := <ident>, 'SET', <expression>

<statement> :=    <assignment>
                | <procedure_call>
                | <if_statement>
                | <while_statement>
                | <do_statement>
                | <for_statement>
                | <compound_statement>

<implementation_part> := <statement>

<function_declaration> := 'FUNC', <ident>, ';', <block>, ';'

<procedure_declaration> := 'PROC', <ident>, ';', <block>, ';'

<specification_part> := [
                            'CONST', <constant_declaration>
                          | 'VAR', <variable_declaration>
                          | <procedure_declaration>
                          | <function_declaration>
                        ]

<block> := <specification_part>, <implementation_part>

<implementation_unit> := 'IMPL', '::', <ident>, <block>, '.'
```

```
<range> := <number>, 'TO', <number>

<array_type> := 'ARR', <ident>, '[', <range>, ']', 'OF', <type>

<range_type> := '[', <range>, ']'

<enumerated_type> := '{', {<ident>, ','}, <ident>, '}'

<basic_type> :=   <ident>
                | <enumerated_type>
                | <range_type>

<type> :=   <basic_type>
          | <array_type>

<variable_declaration> := {<ident>, ':', <ident>, ','}, <ident>, ':', <ident>, ';'

<constant_declaration> := {<ident>, 'IS', <number>, ','}, <ident>, 'IS', <number>,
';'

<formal_parameters> := '(', {<ident>, ';'}, <ident>, ')'

<type_declaration> := 'TYPE', <ident>, '=>', <type>, ';'

<function_interface> := 'FUNC', <ident>, [<formal_parameters>]

<procedure_interface> := 'PROC', <ident>, [<formal_parameters>]

<declaration_unit> :=  'DECL', '::', <ident>,
                      ['CONST', <constant_declaration>],
                      ['VAR', <variable_declaration>],
                      [<type_declaration>],
                      [<procedure_interface>],
                      [<function_interface>],
                      'DECLARATION', 'END'

<basic_program> := 'PROGRAM', <declaration_unit>, <implementation_part>,
'TERMINATE'
```