PascaL0-like EBNF Gramatika

Program

```
program = block “.” ;

block = declaration_part statement_part ;
```

Deklarační Část

```
declaration_part = {( label_declaration_part | constant_declaration_part |
           variable_declaration_part | procedure_declaration_part )};
```

Deklarace Návěští

```
label_declaration_part = “label” label_list “;” ;
```

Deklarace Konstanty

```
constant_declaration_part = “const” type = identifier_list “:=” atom “;” ;
```

Deklarace Proměnné

```
variable_declaration_part = type identifier_list “:=” expression “;” |

                              type identifier_list “;” |

                              type identifier_list “:=” “[“ expression_list “]” “;”  ;
```

Deklarace Procedury

```
procedure_declaration_part =“procedure” ident “;” block
```

Příkazová Část

```
statement_part = “begin”  { statement } “end”

statement = int “:” ( compound_statement | while_do_statement | do_while_statement
          | repeat_statement | for_statement | if_statement | case_statement |
          assignment_statement | procedure_statement | goto_statement |
          ternary_statement | io_statement ) |
          ( compound_statement | while_do_statement | do_while_statement |
          repeat_statement | for_statement | if_statement | case_statement |
          assignment_statement | procedure_statement | goto_statement |
          ternary_statement | io_statement )
```

Příkaz I/O

```
io_statement = ( “write” | “read” ) ident “;”  ;
```

Příkaz Ternární

```
ternary_statement = ident “:=” expression “?” expression “!” expression  “;”  ;
```

Příkaz Přiřazení

```
assignment_statement = ident “:=” expression “;”  ;
```

Příkaz Goto

```
goto_statement = “goto” int “;”   ;
```

Příkaz procedurální

procedure_statement = “call” ident “;” ;

Příkaz Složený

compound_statement = “begin” statement_list “end” ;

Příkaz While Do

while_do_statement = “while” expression “do” statement ;

Příkaz Do While

do_while_statement = “do” statement “while” expression ;

Příkaz Repeat Until

repeat_statement = “repeat” statement “until” expression ;

Příkaz For

for_statement = “for” ident “:=” expression (“to” | “downto”) expression “do” statement ;

Příkaz If

if_statement = “if” expression “then” statement [“else” “then” statement];

Příkaz Case

case_statement = “case” expression “of” case_limb_list “end” ;

case_limb_list = [case_limb] ;

case_limb = case_label_list “:” statement ;

case_label_list = atom { “,” atom } ;

Nízkoúrovňové Definice

ident = ('a' .. 'z' | 'A' .. 'Z') { 'a' .. 'z' | 'A' .. 'Z' | '0' .. '9' | '_' }

type = (“string” | “real” | “integer” | “boolean”) ;

```
label_list = int { “,” int  } ;

identifier_list = ident { “,” ident } ;

expression_list = expression { “,” expression } ;

expression =

              “-“ expression |

             “not” expression |

             expression (“*” | “/”   ) expression |

             expression  ( “+” | “-“ ) expression |

             expression ( “=” | “<>” | “<” | “<=” | “>” | “>=” ) expression

             expression (“and” | “or” ) expression |

             atom |

             “(” expression  “)”;

atom = ( int | float | ident | (“true”  | “false” ) | string  ) ;

string = '\" ('\'\" | ~ ('\"))* '\" ;

float = [0-9] '.' {0-9} | '.' [0-9] ;

int = [0-9] ;
```