Heine-fluch 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 { “,” label } “;” ;

Deklarace Konstanty

constant_declaration_part = “const” type constant_declaration “;” { constant_declaration “;” } ;

constant_declaration = identifier_list “:=” constant ;

Deklarace Proměnné

variable_declaration_part = variable_simple_declaration “;” | variable_paralel_declaration “;” ;

variable_simple_declaration = type identifier_list “:=” expression_list | type identifier_list ;

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

Deklarace Procedury

procedure_declaration_part = procedure_heading “;” procedure_body ;

procedure_body = block ;

procedure_heading = “procedure” identifier ;

Příkazová Část

statement_part = “begin” statement_sequence “end” ;

statement_sequence = statement { “;” statement } ;

statement = label “:” (simple_statement | structured_statement) | (simple_statement |
structured_statement) ;

Příkaz Jednoduchý

simple_statement = (assignment_statement | procedure_statement | goto_statement |
ternary_statement | io_statement) ;

Příkaz I/O

io_statement = (“write” | “read”) identifier ;

Příkaz Ternární
ternary_statement = identifier “:=” expression “?” expression “:” expression ;

Příkaz Přiřazení
assignment_statement = identifier “:=” expression ;

Příkaz Goto
goto_statement = “goto” label ;

Příkaz procedurální
procedure_statement = “call” identifier ;

Příkaz Strukturovaný
structured_statement = (compound_statement | repetitive_statement | conditional_statement) ;

Příkaz Složený
compound_statement = “begin” statement_sequence “end” ;

Příkaz Opakovací
repetetive_statement = (while_do_statement | do_while_statement | repeat_statement | for_statement) ;

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” identifier “:=” expression (“to” | “downto”) expression “do” statement ;

Příkaz Podmínkový
conditional_statement = (if_statement | case_statement) ;

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

Příkaz Case
case_statement = “case” expression “of” case_limb { “;” case_limb } [“;”] “end” ;

case_limb = case_label_list “:” statement ;

Nízkoúrovňové Definice
identifier = ('a' .. 'z' | 'A' .. 'Z') { 'a' .. 'z' | 'A' .. 'Z' | '0' .. '9' | '_' } ;

constant = [sign] (identifier | number) | string ;

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

label = integer_number ;

identifier_list = identifier { “,” identifier } ;

expression_list = expression { “,” expression } ;

case_label_list = constant { “,” constant } ;

expression = simple_expression { relational_operator simple_expression } ;

simple_expression = [sign] term { addition_operator term } ;

term = { “!” } factor { multiplication_operator { “!” } factor } ;

factor = (number | string | identifier | “(“ expression “)”) ;

relational_operator = (“=” | “<>” | “<” | “<=” | “>” | “>=”) ;

addition_operator = (“+” | “-“ | “or”) ;

multiplication_operator = (“*” | “/” | “and”)

string = “ ’ ” string_character { string_character } “ ‘ “ ;

string_character = any-character-except-quote | “ ‘ “ ;

number = (integer_number | real_number) ;

integer_number = digit_sequence ;

real_number = digit_sequence “.” { unsigned_digit_sequence } | digit_sequence ;

digit_sequence = [sign] unsigned_digit_sequence ;

unsigned_digit_sequence = digit { digit } ;

digit = '0' .. '9' ;

sign = (“+” | “-“) ;