Production Rules

program ->   **program id ;**
          declarations
          subprogram_declarations
          compound_statement
          .

identifier_list ->   **id** |
          id , *identifier_list*

declarations ->   **var** *identifier_list* **:** *type* **;** *declarations* |
          λ

type ->   *standard_type* |
          array [num : num] of *standard_type*

standard_type -> **integer** |
          real

subprogram_declarations -> *subprogram_declaration* **;**
              subprogram_declarations |
              λ

subprogram_declaration -> *subprogram_head*
              declarations
              subprogram_declarations
              compound_statement

subprogram_head -> **function id** *arguments* **:** *standard_type* **;** |
          procedure id *arguments* **;**

arguments ->   **(** *parameter_list* **)** |
          λ

parameter_list -> *identifier_list* **:** *type* |
          identifier_list **:** *type* **;** *parameter_list*

compound_statement -> **begin** *optional_statements* **end**

optional_statements -> *statement_list* |
             λ

statement_list ->　　　　*statement |*
　　　　　　　　　　　　*statement **;** statement_list*

statement ->　　　　　*variable **assignop** expression |*
　　　　　　　　　　　　procedure_statement |
　　　　　　　　　　　　compound_statement |
　　　　　　　　　　　　if *expression* **then** *statement* **else** *statement* |
　　　　　　　　　　　　while *expression* **do** *statement* |
　　　　　　　　　　　　read (id) |
　　　　　　　　　　　　write (expression)

variable ->　　　　　　**id** |
　　　　　　　　　　　　id [*expression* **]**

procedure_statement ->　　　　**id** |
　　　　　　　　　　　　　　　id (*expression_list* **)**

expression_list ->　　　*expression |*
　　　　　　　　　　　　*expression **,** expression_list*

expression ->　　　　　*simple_expression |*
　　　　　　　　　　　　simple_expression **relop** *simple_expression*

simple_expression ->　　　　　*term simple_part |*
　　　　　　　　　　　　　　sign term simple_part

simple_part ->　　　　**addop** *term simple_part |*
　　　　　　　　　　　　λ

term ->　　　　　　　*factor term_part*

term_part ->　　　　　**mulop** *factor term_part |*
　　　　　　　　　　　　λ

factor ->　　　　　　　**id** |
　　　　　　　　　　　　id [*expression* **]** |
　　　　　　　　　　　　id (*expression_list* **)** |
　　　　　　　　　　　　num |
　　　　　　　　　　　　(*expression* **)** |
　　　　　　　　　　　　not *factor*

sign ->　　　　　　　　**+** |
　　　　　　　　　　　　-

Lexical Conventions

1. Comments are surrounded by **{** and **}**. They may not contain a {. Comments may appear after any token.

2. Blanks between tokens are optional.

3. Token **id** for identifiers matches a letter followed by letter or digits:
   letter -> [a-zA-Z]
   digit -> [0-9]
   id -> letter (letter | digit)*

The ***** indicates that the choice in the parentheses may be made as many times as you wish.

1. Token **num** matches numbers as follows:
   digits -> digit digit*
   optional_fraction -> . digits | λ
   optional_exponent -> (E (+ | - | λ) digits) | λ
   num -> digits optional_fraction optional_exponent

2. Keywords are reserved.

3. The relational operators (**relop**'s) are:
       =, **<>**, **<**, **<=**, **>=**, and **>**.

4. The **addop**'s are **+**, **-**, and **or**.

5. The **mulop**'s are *****, **/**, **div**, **mod**, and **and**.

6. The lexeme for token **assignop** is **:=**.