Project phase 1 – Scanner and Parser

# DFA for Tiger language scanner:

* Final states in the DFA are highlighted with green color. These are the states which correspond to the tokens produced by the scanner.
* After each token is produced, the scanner is reset to begin from the start state – CHARACTER\_ACCEPT
* The input character column in the following table lists a character or character class. Character classes are represented in uppercase.
  + ALPHA\_NUMERIC – letters [A-Z][a-z] and digits [0-9]
  + DIGIT – digits [0-9]
  + ID\_CHAR – characters that can be part of an identifier letters [A-Z][a-z], digits [0-9] and underscore symbol ( \_ ).
  + SPACE – includes white spaces
  + ANY\_STRING – any character excluding escape characters
  + ESCAPE\_CHAR – includes only the escape symbol
  + ANY – any character in the ASCII character set.
* The DFA is represented in the form of a table in which one row represents a transition: a state followed by the input character and the state to which it moves to:

|  |  |  |
| --- | --- | --- |
| **From State** | **Input character** | **To State** |
| CHARACTER\_ACCEPT | SPACE | CHARACTER\_ACCEPT |
| CHARACTER\_ACCEPT | , | COMMA |
| CHARACTER\_ACCEPT | : | COLON |
| CHARACTER\_ACCEPT | ; | SEMI |
| CHARACTER\_ACCEPT | ( | LPAREN |
| CHARACTER\_ACCEPT | ) | RPAREN |
| CHARACTER\_ACCEPT | ] | RBRACK |
| CHARACTER\_ACCEPT | [ | LBRACK |
| CHARACTER\_ACCEPT | + | PLUS |
| CHARACTER\_ACCEPT | - | MINUS |
| CHARACTER\_ACCEPT | \* | MULT |
| CHARACTER\_ACCEPT | / | DIV |
| CHARACTER\_ACCEPT | = | EQ |
| CHARACTER\_ACCEPT | > | GREATER |
| CHARACTER\_ACCEPT | < | LESSER |
| CHARACTER\_ACCEPT | & | AND |
| CHARACTER\_ACCEPT | | | OR |
| COLON | = | ASSIGN |
| GREATER | = | GREATEREQ |
| LESSER | = | LESSEREQ |
| LESSER | > | NEQ |
| CHARACTER\_ACCEPT | " | STRING\_INTERPRET |
| STRING\_INTERPRET | ANY\_STRING | STRING\_INTERPRET |
| STRING\_INTERPRET | " | STRLIT |
| STRING\_INTERPRET | \ | ESCAPE |
| ESCAPE | \ | STRING\_INTERPRET |
| ESCAPE | " | STRING\_INTERPRET |
| ESCAPE | n | STRING\_INTERPRET |
| ESCAPE | t | STRING\_INTERPRET |
| ESCAPE | ^ | ESCAPE\_C |
| ESCAPE\_C | @ | STRING\_INTERPRET |
| ESCAPE\_C | A | STRING\_INTERPRET |
| ESCAPE\_C | B | STRING\_INTERPRET |
| ESCAPE\_C | C | STRING\_INTERPRET |
| ESCAPE\_C | D | STRING\_INTERPRET |
| ESCAPE\_C | E | STRING\_INTERPRET |
| ESCAPE\_C | F | STRING\_INTERPRET |
| ESCAPE\_C | G | STRING\_INTERPRET |
| ESCAPE\_C | H | STRING\_INTERPRET |
| ESCAPE\_C | I | STRING\_INTERPRET |
| ESCAPE\_C | J | STRING\_INTERPRET |
| ESCAPE\_C | K | STRING\_INTERPRET |
| ESCAPE\_C | L | STRING\_INTERPRET |
| ESCAPE\_C | M | STRING\_INTERPRET |
| ESCAPE\_C | N | STRING\_INTERPRET |
| ESCAPE\_C | O | STRING\_INTERPRET |
| ESCAPE\_C | P | STRING\_INTERPRET |
| ESCAPE\_C | Q | STRING\_INTERPRET |
| ESCAPE\_C | R | STRING\_INTERPRET |
| ESCAPE\_C | S | STRING\_INTERPRET |
| ESCAPE\_C | T | STRING\_INTERPRET |
| ESCAPE\_C | U | STRING\_INTERPRET |
| ESCAPE\_C | V | STRING\_INTERPRET |
| ESCAPE\_C | W | STRING\_INTERPRET |
| ESCAPE\_C | X | STRING\_INTERPRET |
| ESCAPE\_C | Y | STRING\_INTERPRET |
| ESCAPE\_C | Z | STRING\_INTERPRET |
| ESCAPE\_C | [ | STRING\_INTERPRET |
| ESCAPE\_C | \ | STRING\_INTERPRET |
| ESCAPE\_C | ] | STRING\_INTERPRET |
| ESCAPE\_C | ^ | STRING\_INTERPRET |
| ESCAPE\_C | \_ | STRING\_INTERPRET |
| ESCAPE | SPACE | ESCAPE\_WHITE |
| ESCAPE\_WHITE | SPACE | ESCAPE\_WHITE |
| ESCAPE\_WHITE | \ | STRING\_INTERPRET |
| ESCAPE | DIGIT | ESCAPE\_D |
| ESCAPE\_D | DIGIT | ESCAPE\_DD |
| ESCAPE\_DD | DIGIT | STRING\_INTERPRET |
| CHARACTER\_ACCEPT | DIGIT | INTLIT |
| INTLIT | DIGIT | INTLIT |
| CHARACTER\_ACCEPT | a | a |
| a | r | ar |
| ar | r | arr |
| arr | a | arra |
| arra | y | ARRAY |
| CHARACTER\_ACCEPT | b | b |
| b | r | br |
| b | e | be |
| br | e | bre |
| be | g | beg |
| bre | a | brea |
| beg | i | begi |
| brea | k | BREAK |
| begi | n | BEGIN |
| CHARACTER\_ACCEPT | d | d |
| d | o | DO |
| CHARACTER\_ACCEPT | e | e |
| e | l | el |
| e | n | en |
| el | s | els |
| els | e | ELSE |
| en | d | END |
| END | i | endi |
| endi | f | ENDIF |
| END | d | endd |
| endd | o | ENDDO |
| CHARACTER\_ACCEPT | d | d |
| d | o | DO |
| CHARACTER\_ACCEPT | f | f |
| f | o | fo |
| f | u | fu |
| fo | r | FOR |
| fu | n | fun |
| fun | c | func |
| func | t | funct |
| funct | i | functi |
| functi | o | functio |
| functio | n | FUNC |
| CHARACTER\_ACCEPT | i | i |
| i | f | IF |
| i | n | IN |
| CHARACTER\_ACCEPT | l | l |
| l | e | le |
| le | t | LET |
| CHARACTER\_ACCEPT | n | n |
| n | i | ni |
| ni | l | NIL |
| CHARACTER\_ACCEPT | o | o |
| o | f | OF |
| CHARACTER\_ACCEPT | r | r |
| r | e | re |
| re | t | ret |
| ret | u | retu |
| retu | r | retur |
| retur | n | RETURN |
| CHARACTER\_ACCEPT | t | t |
| t | o | TO |
| t | h | th |
| t | y | ty |
| th | e | the |
| ty | p | typ |
| the | n | THEN |
| typ | e | TYPE |
| CHARACTER\_ACCEPT | v | v |
| v | a | va |
| va | r | VAR |
| CHARACTER\_ACCEPT | w | w |
| w | h | wh |
| wh | i | whi |
| whi | l | whil |
| whil | e | WHILE |
| END | ID\_CHAR | ID |
| ID | ID\_CHAR | ID |
| ARRAY | ID\_CHAR | ID |
| BREAK | ID\_CHAR | ID |
| BEGIN | ID\_CHAR | ID |
| DO | ID\_CHAR | ID |
| ELSE | ID\_CHAR | ID |
| END | ID\_CHAR | ID |
| ENDIF | ID\_CHAR | ID |
| ENDDO | ID\_CHAR | ID |
| DO | ID\_CHAR | ID |
| FOR | ID\_CHAR | ID |
| FUNC | ID\_CHAR | ID |
| IF | ID\_CHAR | ID |
| IN | ID\_CHAR | ID |
| LET | ID\_CHAR | ID |
| NIL | ID\_CHAR | ID |
| OF | ID\_CHAR | ID |
| RETURN | ID\_CHAR | ID |
| TO | ID\_CHAR | ID |
| THEN | ID\_CHAR | ID |
| TYPE | ID\_CHAR | ID |
| VAR | ID\_CHAR | ID |
| WHILE | ID\_CHAR | ID |
| CHARACTER\_ACCEPT | ALPHANUMERIC | ID |

# Revised Tiger language grammar

|  |  |
| --- | --- |
| **Non-Terminal** | **Rule Expansion** |
| **# High-level stuff** |  |
| <tiger-program> | LET <declaration-segment> IN <stat-seq> END |
| <declaration-segment> | <type-declaration-list> <var-declaration-list> <funct-declaration-list> |
| <type-declaration-list> | NULL |
| <type-declaration-list> | <type-declaration> <type-declaration-list> |
| <var-declaration-list> | NULL |
| <var-declaration-list> | <var-declaration> <var-declaration-list> |
| <funct-declaration-list> | NULL |
| <funct-declaration-list> | <funct-declaration> <funct-declaration-list> |
| **# Declarations** |  |
| <type-declaration> | TYPE ID EQ <type> SEMI |
| <type> | <type-id> |
| <type> | ARRAY LBRACK INTLIT RBRACK <type-dim> OF <type-id> |
| <type-dim> | NULL |
| <type-dim> | LBRACK INTLIT RBRACK <type-dim> |
| <type-id> | INT |
| <type-id> | STRING |
| <type-id> | ID |
| **# Variables** |  |
| <var-declaration> | VAR <id-list> COLON <type-id> <optional-init> SEMI |
| <id-list> | ID <id-list-tail> |
| <id-list-tail> | NULL |
| <id-list-tail> | COMMA ID <id-list-tail> |
| <optional-init> | NULL |
| <optional-init> | ASSIGN <const> |
| **# Functions** |  |
| <funct-declaration> | FUNC ID LPAREN <param-list> RPAREN <ret-type> BEGIN <stat-seq> END SEMI |
| <param-list> | NULL |
| <param-list> | <param> <param-list-tail> |
| <param-list-tail> | NULL |
| <param-list-tail> | COMMA <param> <param-list-tail> |
| <ret-type> | NULL |
| <ret-type> | COLON <type-id> |
| <param> | ID COLON <type-id> |
| **# Statements** |  |
| <stat-seq> | <stat> <stat-seq-tail> |
| <stat-seq-tail> | <stat> <stat-seq-tail> |
| <stat-seq-tail> | NULL |
| <stat> | IF <expr> THEN <stat-seq> <else-part> ENDIF SEMI |
| <else-part> | ELSE <stat-seq> |
| <else-part> | NULL |
| <stat> | WHILE <expr> DO <stat-seq> ENDDO SEMI |
| <stat> | FOR ID ASSIGN <expr> TO <expr> DO <stat-seq> ENDDO SEMI |
| <stat> | BREAK SEMI |
| <stat> | RETURN <expr> SEMI |
| <stat> | ID <stat-after-id> |
| <stat-after-id> | LPAREN <expr-list> RPAREN SEMI |
| <stat-after-id> | <lvalue-tail> ASSIGN <rvalue> SEMI |
| <rvalue> | <expr-no-lvalue> |
| <rvalue> | ID <expr-or-func> |
| <expr-or-func> | LPAREN <expr-list> RPAREN |
| <expr-or-func> | <expr-after-id> |
| **# Expressions** |  |
| <expr> | <or-term> <expr-tail> |
| <expr-no-lvalue> | <or-term-no-lvalue> <expr-tail> |
| <expr-after-id> | <or-term-after-id> <expr-tail> |
| <expr-tail> | NULL |
| <expr-tail> | OR <or-term> <expr-tail> |
| <or-term> | <and-term> <or-term-tail> |
| <or-term-no-lvalue> | <and-term-no-lvalue> <or-term-tail> |
| <or-term-after-id> | <and-term-after-id> <or-term-tail> |
| <or-term-tail> | NULL |
| <or-term-tail> | AND <and-term> <or-term-tail> |
| <and-term> | <comp-term> <and-term-tail> |
| <and-term-no-lvalue> | <comp-term-no-lvalue> <and-term-tail> |
| <and-term-after-id> | <comp-term-after-id> <and-term-tail> |
| <and-term-tail> | NULL |
| <and-term-tail> | <comp-op> <comp-term> |
| <comp-op> | EQ |
| <comp-op> | NEQ |
| <comp-op> | LESSER |
| <comp-op> | GREATER |
| <comp-op> | LESSEREQ |
| <comp-op> | GREATEREQ |
| <comp-term> | <term> <comp-term-tail> |
| <comp-term-no-lvalue> | <term-no-lvalue> <comp-term-tail> |
| <comp-term-after-id> | <term-after-id> <comp-term-tail> |
| <comp-term-tail> | NULL |
| <comp-term-tail> | <add-op> <term> <comp-term-tail> |
| <add-op> | PLUS |
| <add-op> | MINUS |
| <term> | <factor> <term-tail> |
| <term-no-lvalue> | <factor-no-lvalue> <term-tail> |
| <term-after-id> | <lvalue-tail> <term-tail> |
| <term-tail> | NULL |
| <term-tail> | <mult-op> <factor> <term-tail> |
| <mult-op> | MULT |
| <mult-op> | DIV |
| <factor> | <factor-no-lvalue> |
| <factor> | <lvalue> |
| <factor-no-lvalue> | <const> |
| <factor-no-lvalue> | MINUS <factor> |
| <factor-no-lvalue> | LPAREN <expr> RPAREN |
| <const> | INTLIT |
| <const> | STRLIT |
| <const> | NIL |
| <expr-list> | NULL |
| <expr-list> | <expr> <expr-list-tail> |
| <expr-list-tail> | COMMA <expr> <expr-list-tail> |
| <expr-list-tail> | NULL |
| <lvalue> | ID <lvalue-tail> |
| <lvalue-tail> | LBRACK <expr> RBRACK <lvalue-tail> |
| <lvalue-tail> | NULL |

# Parser table construction

First and follow sets of the non-terminals in the grammar are as follows:

|  |  |  |
| --- | --- | --- |
| **Symbol** | **First set** | **Follow Set** |
| <add-op> | PLUS, MINUS | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| <and-term-after-id> | PLUS, GREATER, LESSER, NEQ, LBRACK, LESSEREQ, DIV, MULT, EQ, MINUS, NULL, GREATEREQ | AND, SEMI, OR |
| <and-term-no-lvalue> | STRLIT, INTLIT, NIL, MINUS, LPAREN | AND, SEMI, OR |
| <and-term-tail> | GREATER, LESSER, NEQ, LESSEREQ, EQ, NULL, GREATEREQ | RBRACK, RPAREN, DO, THEN, AND, TO, COMMA, SEMI, OR |
| <and-term> | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN | RBRACK, RPAREN, DO, THEN, AND, TO, COMMA, SEMI, OR |
| <comp-op> | GREATER, LESSER, NEQ, LESSEREQ, EQ, GREATEREQ | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| <comp-term-after-id> | PLUS, LBRACK, DIV, MULT, MINUS, NULL | GREATER, LESSER, NEQ, LESSEREQ, EQ, AND, SEMI, OR, GREATEREQ |
| <comp-term-no-lvalue> | STRLIT, INTLIT, NIL, MINUS, LPAREN | GREATER, LESSER, NEQ, LESSEREQ, EQ, AND, SEMI, OR, GREATEREQ |
| <comp-term-tail> | PLUS, MINUS, NULL | RBRACK, RPAREN, GREATER, LESSER, NEQ, THEN, TO, COMMA, OR, DO, LESSEREQ, EQ, AND, SEMI, GREATEREQ |
| <comp-term> | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN | RBRACK, RPAREN, NEQ, LESSER, GREATER, THEN, TO, COMMA, OR, DO, LESSEREQ, EQ, AND, SEMI, GREATEREQ |
| <const> | STRLIT, INTLIT, NIL | RBRACK, RPAREN, GREATER, LESSER, NEQ, THEN, TO, COMMA, OR, DO, PLUS, LESSEREQ, EQ, MINUS, MULT, DIV, AND, SEMI, GREATEREQ |
| <declaration-segment> | VAR, FUNC, NULL, TYPE | IN |
| <else-part> | NULL, ELSE | ENDIF |
| <expr-after-id> | LBRACK, NEQ, LESSER, GREATER, NULL, OR, PLUS, LESSEREQ, MULT, MINUS, EQ, DIV, AND, GREATEREQ | SEMI |
| <expr-list-tail> | COMMA, NULL | RPAREN |
| <expr-list> | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN, NULL | RPAREN |
| <expr-no-lvalue> | STRLIT, INTLIT, NIL, MINUS, LPAREN | SEMI |
| <expr-or-func> | LBRACK, NEQ, LESSER, GREATER, NULL, OR, PLUS, LESSEREQ, MULT, DIV, MINUS, EQ, AND, LPAREN, GREATEREQ | SEMI |
| <expr-tail> | NULL, OR | RBRACK, RPAREN, DO, THEN, TO, COMMA, SEMI |
| <expr> | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN | RBRACK, RPAREN, DO, THEN, TO, COMMA, SEMI |
| <factor-no-lvalue> | STRLIT, INTLIT, NIL, MINUS, LPAREN | RBRACK, RPAREN, GREATER, LESSER, NEQ, THEN, TO, COMMA, OR, DO, PLUS, LESSEREQ, EQ, DIV, MULT, MINUS, AND, SEMI, GREATEREQ |
| <factor> | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN | RBRACK, RPAREN, NEQ, GREATER, LESSER, THEN, TO, COMMA, OR, DO, PLUS, LESSEREQ, EQ, MINUS, MULT, DIV, AND, SEMI, GREATEREQ |
| <funct-declaration-list> | FUNC, NULL | IN |
| <funct-declaration> | FUNC | IN, FUNC |
| <id-list-tail> | COMMA, NULL | COLON |
| <id-list> | ID | COLON |
| <lvalue-tail> | LBRACK, NULL | RBRACK, RPAREN, GREATER, LESSER, NEQ, THEN, TO, COMMA, OR, ASSIGN, DO, PLUS, LESSEREQ, EQ, DIV, MINUS, MULT, AND, SEMI, GREATEREQ |
| <lvalue> | ID | RBRACK, RPAREN, NEQ, GREATER, LESSER, THEN, TO, COMMA, OR, DO, PLUS, LESSEREQ, EQ, DIV, MULT, MINUS, AND, SEMI, GREATEREQ |
| <mult-op> | DIV, MULT | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| <optional-init> | NULL, ASSIGN | SEMI |
| <or-term-after-id> | LBRACK, NEQ, LESSER, GREATER, NULL, PLUS, LESSEREQ, MINUS, EQ, MULT, DIV, AND, GREATEREQ | SEMI, OR |
| <or-term-no-lvalue> | STRLIT, INTLIT, NIL, MINUS, LPAREN | SEMI, OR |
| <or-term-tail> | AND, NULL | RBRACK, RPAREN, DO, THEN, TO, COMMA, SEMI, OR |
| <or-term> | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN | RBRACK, RPAREN, DO, THEN, TO, COMMA, SEMI, OR |
| <param-list-tail> | COMMA, NULL | RPAREN |
| <param-list> | ID, NULL | RPAREN |
| <param> | ID | RPAREN, COMMA |
| <ret-type> | COLON, NULL | BEGIN |
| <rvalue> | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN | SEMI |
| <stat-after-id> | LBRACK, LPAREN, ASSIGN | ENDIF, FOR, ENDDO, WHILE, ID, END, BREAK, ELSE, IF, RETURN |
| <stat-seq-tail> | FOR, WHILE, ID, BREAK, NULL, IF, RETURN | ENDIF, ENDDO, END, ELSE |
| <stat-seq> | FOR, WHILE, ID, BREAK, IF, RETURN | ENDIF, ENDDO, END, ELSE |
| <stat> | FOR, WHILE, ID, BREAK, IF, RETURN | ENDIF, FOR, ENDDO, WHILE, ID, END, BREAK, ELSE, IF, RETURN |
| <term-after-id> | LBRACK, DIV, MULT, NULL | PLUS, GREATER, LESSER, NEQ, LESSEREQ, MINUS, EQ, AND, SEMI, OR, GREATEREQ |
| <term-no-lvalue> | STRLIT, INTLIT, NIL, MINUS, LPAREN | PLUS, GREATER, LESSER, NEQ, LESSEREQ, MINUS, EQ, AND, SEMI, OR, GREATEREQ |
| <term-tail> | DIV, MULT, NULL | RBRACK, RPAREN, NEQ, LESSER, GREATER, THEN, TO, COMMA, OR, DO, PLUS, LESSEREQ, EQ, MINUS, AND, SEMI, GREATEREQ |
| <term> | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN | RBRACK, RPAREN, NEQ, GREATER, LESSER, THEN, TO, COMMA, OR, DO, PLUS, LESSEREQ, EQ, MINUS, AND, SEMI, GREATEREQ |
| <tiger-program> | LET | $ |
| <type-declaration-list> | NULL, TYPE | VAR, IN, FUNC |
| <type-declaration> | TYPE | VAR, IN, FUNC, TYPE |
| <type-dim> | LBRACK, NULL | OF |
| <type-id> | INT, ID, STRING | RPAREN, COMMA, SEMI, BEGIN, ASSIGN |
| <type> | INT, ID, ARRAY, STRING | SEMI |
| <var-declaration-list> | VAR, NULL | IN, FUNC |
| <var-declaration> | VAR | VAR, IN, FUNC |
| AND | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| ARRAY | null | LBRACK |
| ASSIGN | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| BEGIN | null | FOR, WHILE, ID, BREAK, IF, RETURN |
| BREAK | null | SEMI |
| COLON | null | INT, ID, STRING |
| COMMA | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| DIV | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| DO | null | FOR, WHILE, ID, BREAK, IF, RETURN |
| ELSE | null | FOR, WHILE, ID, BREAK, IF, RETURN |
| END | null | $, SEMI |
| ENDDO | null | SEMI |
| ENDIF | null | SEMI |
| EQ | null | STRLIT, INT, INTLIT, NIL, MINUS, ID, ARRAY, LPAREN, STRING |
| FOR | null | ID |
| FUNC | null | ID |
| GREATER | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| GREATEREQ | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| ID | null | COLON, RBRACK, RPAREN, NEQ, GREATER, LESSER, LBRACK, THEN, TO, COMMA, BEGIN, OR, ASSIGN, DO, PLUS, LESSEREQ, DIV, MULT, MINUS, EQ, AND, SEMI, LPAREN, GREATEREQ |
| IF | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| IN | null | FOR, WHILE, ID, BREAK, IF, RETURN |
| INT | null | RPAREN, COMMA, SEMI, BEGIN, ASSIGN |
| INTLIT | null | RBRACK, RPAREN, GREATER, LESSER, NEQ, THEN, TO, COMMA, OR, DO, PLUS, LESSEREQ, EQ, DIV, MULT, MINUS, AND, SEMI, GREATEREQ |
| LBRACK | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| LESSER | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| LESSEREQ | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| LET | null | VAR, IN, FUNC, TYPE |
| LPAREN | null | STRLIT, RPAREN, INTLIT, NIL, MINUS, ID, LPAREN |
| MINUS | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| MULT | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| NEQ | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| NIL | null | RBRACK, RPAREN, GREATER, LESSER, NEQ, THEN, TO, COMMA, OR, DO, PLUS, LESSEREQ, EQ, DIV, MULT, MINUS, AND, SEMI, GREATEREQ |
| NULL | null | RBRACK, TO, ELSE, DO, LESSEREQ, MINUS, MULT, AND, OF, SEMI, COLON, RPAREN, GREATER, LESSER, NEQ, THEN, IN, COMMA, FUNC, BEGIN, OR, ASSIGN, ENDIF, PLUS, ENDDO, VAR, EQ, DIV, END, GREATEREQ |
| OF | null | INT, ID, STRING |
| OR | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| PLUS | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| RBRACK | null | RBRACK, RPAREN, GREATER, LESSER, NEQ, LBRACK, THEN, TO, COMMA, OR, ASSIGN, DO, PLUS, LESSEREQ, EQ, DIV, MULT, MINUS, AND, OF, SEMI, GREATEREQ |
| RETURN | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| RPAREN | null | COLON, RBRACK, RPAREN, GREATER, LESSER, NEQ, THEN, TO, COMMA, BEGIN, OR, DO, PLUS, LESSEREQ, EQ, DIV, MULT, MINUS, AND, SEMI, GREATEREQ |
| SEMI | null | IN, WHILE, FUNC, ELSE, RETURN, ENDIF, FOR, VAR, ENDDO, END, ID, BREAK, TYPE, IF |
| STRING | null | RPAREN, COMMA, SEMI, BEGIN, ASSIGN |
| STRLIT | null | RBRACK, RPAREN, GREATER, LESSER, NEQ, THEN, TO, COMMA, OR, DO, PLUS, LESSEREQ, EQ, DIV, MULT, MINUS, AND, SEMI, GREATEREQ |
| THEN | null | FOR, WHILE, ID, BREAK, IF, RETURN |
| TO | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |
| TYPE | null | ID |
| VAR | null | ID |
| WHILE | null | STRLIT, INTLIT, NIL, MINUS, ID, LPAREN |

## Parser Table:

The parser table is represented in the way it would be indexed.

|  |  |  |
| --- | --- | --- |
| **Row index** | **Column index** | **Expansion rule** |
| <add-op> | MINUS | <add-op> -> MINUS |
| <add-op> | PLUS | <add-op> -> PLUS |
| <and-term-after-id> | AND | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | DIV | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | EQ | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | GREATER | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | GREATEREQ | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | LBRACK | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | LESSER | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | LESSEREQ | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | MINUS | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | MULT | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | NEQ | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | OR | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | PLUS | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-after-id> | SEMI | <and-term-after-id> -> <comp-term-after-id> <and-term-tail> |
| <and-term-no-lvalue> | INTLIT | <and-term-no-lvalue> -> <comp-term-no-lvalue> <and-term-tail> |
| <and-term-no-lvalue> | LPAREN | <and-term-no-lvalue> -> <comp-term-no-lvalue> <and-term-tail> |
| <and-term-no-lvalue> | MINUS | <and-term-no-lvalue> -> <comp-term-no-lvalue> <and-term-tail> |
| <and-term-no-lvalue> | NIL | <and-term-no-lvalue> -> <comp-term-no-lvalue> <and-term-tail> |
| <and-term-no-lvalue> | STRLIT | <and-term-no-lvalue> -> <comp-term-no-lvalue> <and-term-tail> |
| <and-term-tail> | AND | <and-term-tail> -> NULL |
| <and-term-tail> | COMMA | <and-term-tail> -> NULL |
| <and-term-tail> | DO | <and-term-tail> -> NULL |
| <and-term-tail> | EQ | <and-term-tail> -> <comp-op> <comp-term> |
| <and-term-tail> | GREATER | <and-term-tail> -> <comp-op> <comp-term> |
| <and-term-tail> | GREATEREQ | <and-term-tail> -> <comp-op> <comp-term> |
| <and-term-tail> | LESSER | <and-term-tail> -> <comp-op> <comp-term> |
| <and-term-tail> | LESSEREQ | <and-term-tail> -> <comp-op> <comp-term> |
| <and-term-tail> | NEQ | <and-term-tail> -> <comp-op> <comp-term> |
| <and-term-tail> | OR | <and-term-tail> -> NULL |
| <and-term-tail> | RBRACK | <and-term-tail> -> NULL |
| <and-term-tail> | RPAREN | <and-term-tail> -> NULL |
| <and-term-tail> | SEMI | <and-term-tail> -> NULL |
| <and-term-tail> | THEN | <and-term-tail> -> NULL |
| <and-term-tail> | TO | <and-term-tail> -> NULL |
| <and-term> | ID | <and-term> -> <comp-term> <and-term-tail> |
| <and-term> | INTLIT | <and-term> -> <comp-term> <and-term-tail> |
| <and-term> | LPAREN | <and-term> -> <comp-term> <and-term-tail> |
| <and-term> | MINUS | <and-term> -> <comp-term> <and-term-tail> |
| <and-term> | NIL | <and-term> -> <comp-term> <and-term-tail> |
| <and-term> | STRLIT | <and-term> -> <comp-term> <and-term-tail> |
| <comp-op> | EQ | <comp-op> -> EQ |
| <comp-op> | GREATER | <comp-op> -> GREATER |
| <comp-op> | GREATEREQ | <comp-op> -> GREATEREQ |
| <comp-op> | LESSER | <comp-op> -> LESSER |
| <comp-op> | LESSEREQ | <comp-op> -> LESSEREQ |
| <comp-op> | NEQ | <comp-op> -> NEQ |
| <comp-term-after-id> | AND | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | DIV | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | EQ | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | GREATER | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | GREATEREQ | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | LBRACK | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | LESSER | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | LESSEREQ | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | MINUS | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | MULT | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | NEQ | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | OR | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | PLUS | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-after-id> | SEMI | <comp-term-after-id> -> <term-after-id> <comp-term-tail> |
| <comp-term-no-lvalue> | INTLIT | <comp-term-no-lvalue> -> <term-no-lvalue> <comp-term-tail> |
| <comp-term-no-lvalue> | LPAREN | <comp-term-no-lvalue> -> <term-no-lvalue> <comp-term-tail> |
| <comp-term-no-lvalue> | MINUS | <comp-term-no-lvalue> -> <term-no-lvalue> <comp-term-tail> |
| <comp-term-no-lvalue> | NIL | <comp-term-no-lvalue> -> <term-no-lvalue> <comp-term-tail> |
| <comp-term-no-lvalue> | STRLIT | <comp-term-no-lvalue> -> <term-no-lvalue> <comp-term-tail> |
| <comp-term-tail> | AND | <comp-term-tail> -> NULL |
| <comp-term-tail> | COMMA | <comp-term-tail> -> NULL |
| <comp-term-tail> | DO | <comp-term-tail> -> NULL |
| <comp-term-tail> | EQ | <comp-term-tail> -> NULL |
| <comp-term-tail> | GREATER | <comp-term-tail> -> NULL |
| <comp-term-tail> | GREATEREQ | <comp-term-tail> -> NULL |
| <comp-term-tail> | LESSER | <comp-term-tail> -> NULL |
| <comp-term-tail> | LESSEREQ | <comp-term-tail> -> NULL |
| <comp-term-tail> | MINUS | <comp-term-tail> -> <add-op> <term> <comp-term-tail> |
| <comp-term-tail> | NEQ | <comp-term-tail> -> NULL |
| <comp-term-tail> | OR | <comp-term-tail> -> NULL |
| <comp-term-tail> | PLUS | <comp-term-tail> -> <add-op> <term> <comp-term-tail> |
| <comp-term-tail> | RBRACK | <comp-term-tail> -> NULL |
| <comp-term-tail> | RPAREN | <comp-term-tail> -> NULL |
| <comp-term-tail> | SEMI | <comp-term-tail> -> NULL |
| <comp-term-tail> | THEN | <comp-term-tail> -> NULL |
| <comp-term-tail> | TO | <comp-term-tail> -> NULL |
| <comp-term> | ID | <comp-term> -> <term> <comp-term-tail> |
| <comp-term> | INTLIT | <comp-term> -> <term> <comp-term-tail> |
| <comp-term> | LPAREN | <comp-term> -> <term> <comp-term-tail> |
| <comp-term> | MINUS | <comp-term> -> <term> <comp-term-tail> |
| <comp-term> | NIL | <comp-term> -> <term> <comp-term-tail> |
| <comp-term> | STRLIT | <comp-term> -> <term> <comp-term-tail> |
| <const> | INTLIT | <const> -> INTLIT |
| <const> | NIL | <const> -> NIL |
| <const> | STRLIT | <const> -> STRLIT |
| <declaration-segment> | FUNC | <declaration-segment> -> <type-declaration-list> <var-declaration-list> <funct-declaration-list> |
| <declaration-segment> | IN | <declaration-segment> -> <type-declaration-list> <var-declaration-list> <funct-declaration-list> |
| <declaration-segment> | TYPE | <declaration-segment> -> <type-declaration-list> <var-declaration-list> <funct-declaration-list> |
| <declaration-segment> | VAR | <declaration-segment> -> <type-declaration-list> <var-declaration-list> <funct-declaration-list> |
| <else-part> | ELSE | <else-part> -> ELSE <stat-seq> |
| <else-part> | ENDIF | <else-part> -> NULL |
| <expr-after-id> | AND | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | DIV | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | EQ | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | GREATER | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | GREATEREQ | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | LBRACK | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | LESSER | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | LESSEREQ | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | MINUS | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | MULT | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | NEQ | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | OR | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | PLUS | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-after-id> | SEMI | <expr-after-id> -> <or-term-after-id> <expr-tail> |
| <expr-list-tail> | COMMA | <expr-list-tail> -> COMMA <expr> <expr-list-tail> |
| <expr-list-tail> | RPAREN | <expr-list-tail> -> NULL |
| <expr-list> | ID | <expr-list> -> <expr> <expr-list-tail> |
| <expr-list> | INTLIT | <expr-list> -> <expr> <expr-list-tail> |
| <expr-list> | LPAREN | <expr-list> -> <expr> <expr-list-tail> |
| <expr-list> | MINUS | <expr-list> -> <expr> <expr-list-tail> |
| <expr-list> | NIL | <expr-list> -> <expr> <expr-list-tail> |
| <expr-list> | RPAREN | <expr-list> -> NULL |
| <expr-list> | STRLIT | <expr-list> -> <expr> <expr-list-tail> |
| <expr-no-lvalue> | INTLIT | <expr-no-lvalue> -> <or-term-no-lvalue> <expr-tail> |
| <expr-no-lvalue> | LPAREN | <expr-no-lvalue> -> <or-term-no-lvalue> <expr-tail> |
| <expr-no-lvalue> | MINUS | <expr-no-lvalue> -> <or-term-no-lvalue> <expr-tail> |
| <expr-no-lvalue> | NIL | <expr-no-lvalue> -> <or-term-no-lvalue> <expr-tail> |
| <expr-no-lvalue> | STRLIT | <expr-no-lvalue> -> <or-term-no-lvalue> <expr-tail> |
| <expr-or-func> | AND | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | DIV | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | EQ | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | GREATER | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | GREATEREQ | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | LBRACK | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | LESSER | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | LESSEREQ | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | LPAREN | <expr-or-func> -> LPAREN <expr-list> RPAREN |
| <expr-or-func> | MINUS | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | MULT | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | NEQ | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | OR | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | PLUS | <expr-or-func> -> <expr-after-id> |
| <expr-or-func> | SEMI | <expr-or-func> -> <expr-after-id> |
| <expr-tail> | COMMA | <expr-tail> -> NULL |
| <expr-tail> | DO | <expr-tail> -> NULL |
| <expr-tail> | OR | <expr-tail> -> OR <or-term> <expr-tail> |
| <expr-tail> | RBRACK | <expr-tail> -> NULL |
| <expr-tail> | RPAREN | <expr-tail> -> NULL |
| <expr-tail> | SEMI | <expr-tail> -> NULL |
| <expr-tail> | THEN | <expr-tail> -> NULL |
| <expr-tail> | TO | <expr-tail> -> NULL |
| <expr> | ID | <expr> -> <or-term> <expr-tail> |
| <expr> | INTLIT | <expr> -> <or-term> <expr-tail> |
| <expr> | LPAREN | <expr> -> <or-term> <expr-tail> |
| <expr> | MINUS | <expr> -> <or-term> <expr-tail> |
| <expr> | NIL | <expr> -> <or-term> <expr-tail> |
| <expr> | STRLIT | <expr> -> <or-term> <expr-tail> |
| <factor-no-lvalue> | INTLIT | <factor-no-lvalue> -> <const> |
| <factor-no-lvalue> | LPAREN | <factor-no-lvalue> -> LPAREN <expr> RPAREN |
| <factor-no-lvalue> | MINUS | <factor-no-lvalue> -> MINUS <factor> |
| <factor-no-lvalue> | NIL | <factor-no-lvalue> -> <const> |
| <factor-no-lvalue> | STRLIT | <factor-no-lvalue> -> <const> |
| <factor> | ID | <factor> -> <lvalue> |
| <factor> | INTLIT | <factor> -> <factor-no-lvalue> |
| <factor> | LPAREN | <factor> -> <factor-no-lvalue> |
| <factor> | MINUS | <factor> -> <factor-no-lvalue> |
| <factor> | NIL | <factor> -> <factor-no-lvalue> |
| <factor> | STRLIT | <factor> -> <factor-no-lvalue> |
| <funct-declaration-list> | FUNC | <funct-declaration-list> -> <funct-declaration> <funct-declaration-list> |
| <funct-declaration-list> | IN | <funct-declaration-list> -> NULL |
| <funct-declaration> | FUNC | <funct-declaration> -> FUNC ID LPAREN <param-list> RPAREN <ret-type> BEGIN <stat-seq> END SEMI |
| <id-list-tail> | COLON | <id-list-tail> -> NULL |
| <id-list-tail> | COMMA | <id-list-tail> -> COMMA ID <id-list-tail> |
| <id-list> | ID | <id-list> -> ID <id-list-tail> |
| <lvalue-tail> | AND | <lvalue-tail> -> NULL |
| <lvalue-tail> | ASSIGN | <lvalue-tail> -> NULL |
| <lvalue-tail> | COMMA | <lvalue-tail> -> NULL |
| <lvalue-tail> | DIV | <lvalue-tail> -> NULL |
| <lvalue-tail> | DO | <lvalue-tail> -> NULL |
| <lvalue-tail> | EQ | <lvalue-tail> -> NULL |
| <lvalue-tail> | GREATER | <lvalue-tail> -> NULL |
| <lvalue-tail> | GREATEREQ | <lvalue-tail> -> NULL |
| <lvalue-tail> | LBRACK | <lvalue-tail> -> LBRACK <expr> RBRACK <lvalue-tail> |
| <lvalue-tail> | LESSER | <lvalue-tail> -> NULL |
| <lvalue-tail> | LESSEREQ | <lvalue-tail> -> NULL |
| <lvalue-tail> | MINUS | <lvalue-tail> -> NULL |
| <lvalue-tail> | MULT | <lvalue-tail> -> NULL |
| <lvalue-tail> | NEQ | <lvalue-tail> -> NULL |
| <lvalue-tail> | OR | <lvalue-tail> -> NULL |
| <lvalue-tail> | PLUS | <lvalue-tail> -> NULL |
| <lvalue-tail> | RBRACK | <lvalue-tail> -> NULL |
| <lvalue-tail> | RPAREN | <lvalue-tail> -> NULL |
| <lvalue-tail> | SEMI | <lvalue-tail> -> NULL |
| <lvalue-tail> | THEN | <lvalue-tail> -> NULL |
| <lvalue-tail> | TO | <lvalue-tail> -> NULL |
| <lvalue> | ID | <lvalue> -> ID <lvalue-tail> |
| <mult-op> | DIV | <mult-op> -> DIV |
| <mult-op> | MULT | <mult-op> -> MULT |
| <optional-init> | ASSIGN | <optional-init> -> ASSIGN <const> |
| <optional-init> | SEMI | <optional-init> -> NULL |
| <or-term-after-id> | AND | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | DIV | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | EQ | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | GREATER | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | GREATEREQ | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | LBRACK | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | LESSER | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | LESSEREQ | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | MINUS | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | MULT | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | NEQ | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | OR | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | PLUS | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-after-id> | SEMI | <or-term-after-id> -> <and-term-after-id> <or-term-tail> |
| <or-term-no-lvalue> | INTLIT | <or-term-no-lvalue> -> <and-term-no-lvalue> <or-term-tail> |
| <or-term-no-lvalue> | LPAREN | <or-term-no-lvalue> -> <and-term-no-lvalue> <or-term-tail> |
| <or-term-no-lvalue> | MINUS | <or-term-no-lvalue> -> <and-term-no-lvalue> <or-term-tail> |
| <or-term-no-lvalue> | NIL | <or-term-no-lvalue> -> <and-term-no-lvalue> <or-term-tail> |
| <or-term-no-lvalue> | STRLIT | <or-term-no-lvalue> -> <and-term-no-lvalue> <or-term-tail> |
| <or-term-tail> | AND | <or-term-tail> -> AND <and-term> <or-term-tail> |
| <or-term-tail> | COMMA | <or-term-tail> -> NULL |
| <or-term-tail> | DO | <or-term-tail> -> NULL |
| <or-term-tail> | OR | <or-term-tail> -> NULL |
| <or-term-tail> | RBRACK | <or-term-tail> -> NULL |
| <or-term-tail> | RPAREN | <or-term-tail> -> NULL |
| <or-term-tail> | SEMI | <or-term-tail> -> NULL |
| <or-term-tail> | THEN | <or-term-tail> -> NULL |
| <or-term-tail> | TO | <or-term-tail> -> NULL |
| <or-term> | ID | <or-term> -> <and-term> <or-term-tail> |
| <or-term> | INTLIT | <or-term> -> <and-term> <or-term-tail> |
| <or-term> | LPAREN | <or-term> -> <and-term> <or-term-tail> |
| <or-term> | MINUS | <or-term> -> <and-term> <or-term-tail> |
| <or-term> | NIL | <or-term> -> <and-term> <or-term-tail> |
| <or-term> | STRLIT | <or-term> -> <and-term> <or-term-tail> |
| <param-list-tail> | COMMA | <param-list-tail> -> COMMA <param> <param-list-tail> |
| <param-list-tail> | RPAREN | <param-list-tail> -> NULL |
| <param-list> | ID | <param-list> -> <param> <param-list-tail> |
| <param-list> | RPAREN | <param-list> -> NULL |
| <param> | ID | <param> -> ID COLON <type-id> |
| <ret-type> | BEGIN | <ret-type> -> NULL |
| <ret-type> | COLON | <ret-type> -> COLON <type-id> |
| <rvalue> | ID | <rvalue> -> ID <expr-or-func> |
| <rvalue> | INTLIT | <rvalue> -> <expr-no-lvalue> |
| <rvalue> | LPAREN | <rvalue> -> <expr-no-lvalue> |
| <rvalue> | MINUS | <rvalue> -> <expr-no-lvalue> |
| <rvalue> | NIL | <rvalue> -> <expr-no-lvalue> |
| <rvalue> | STRLIT | <rvalue> -> <expr-no-lvalue> |
| <stat-after-id> | ASSIGN | <stat-after-id> -> <lvalue-tail> ASSIGN <rvalue> SEMI |
| <stat-after-id> | LBRACK | <stat-after-id> -> <lvalue-tail> ASSIGN <rvalue> SEMI |
| <stat-after-id> | LPAREN | <stat-after-id> -> LPAREN <expr-list> RPAREN SEMI |
| <stat-seq-tail> | BREAK | <stat-seq-tail> -> <stat> <stat-seq-tail> |
| <stat-seq-tail> | ELSE | <stat-seq-tail> -> NULL |
| <stat-seq-tail> | END | <stat-seq-tail> -> NULL |
| <stat-seq-tail> | ENDDO | <stat-seq-tail> -> NULL |
| <stat-seq-tail> | ENDIF | <stat-seq-tail> -> NULL |
| <stat-seq-tail> | FOR | <stat-seq-tail> -> <stat> <stat-seq-tail> |
| <stat-seq-tail> | ID | <stat-seq-tail> -> <stat> <stat-seq-tail> |
| <stat-seq-tail> | IF | <stat-seq-tail> -> <stat> <stat-seq-tail> |
| <stat-seq-tail> | RETURN | <stat-seq-tail> -> <stat> <stat-seq-tail> |
| <stat-seq-tail> | WHILE | <stat-seq-tail> -> <stat> <stat-seq-tail> |
| <stat-seq> | BREAK | <stat-seq> -> <stat> <stat-seq-tail> |
| <stat-seq> | FOR | <stat-seq> -> <stat> <stat-seq-tail> |
| <stat-seq> | ID | <stat-seq> -> <stat> <stat-seq-tail> |
| <stat-seq> | IF | <stat-seq> -> <stat> <stat-seq-tail> |
| <stat-seq> | RETURN | <stat-seq> -> <stat> <stat-seq-tail> |
| <stat-seq> | WHILE | <stat-seq> -> <stat> <stat-seq-tail> |
| <stat> | BREAK | <stat> -> BREAK SEMI |
| <stat> | FOR | <stat> -> FOR ID ASSIGN <expr> TO <expr> DO <stat-seq> ENDDO SEMI |
| <stat> | ID | <stat> -> ID <stat-after-id> |
| <stat> | IF | <stat> -> IF <expr> THEN <stat-seq> <else-part> ENDIF SEMI |
| <stat> | RETURN | <stat> -> RETURN <expr> SEMI |
| <stat> | WHILE | <stat> -> WHILE <expr> DO <stat-seq> ENDDO SEMI |
| <term-after-id> | AND | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | DIV | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | EQ | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | GREATER | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | GREATEREQ | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | LBRACK | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | LESSER | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | LESSEREQ | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | MINUS | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | MULT | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | NEQ | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | OR | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | PLUS | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-after-id> | SEMI | <term-after-id> -> <lvalue-tail> <term-tail> |
| <term-no-lvalue> | INTLIT | <term-no-lvalue> -> <factor-no-lvalue> <term-tail> |
| <term-no-lvalue> | LPAREN | <term-no-lvalue> -> <factor-no-lvalue> <term-tail> |
| <term-no-lvalue> | MINUS | <term-no-lvalue> -> <factor-no-lvalue> <term-tail> |
| <term-no-lvalue> | NIL | <term-no-lvalue> -> <factor-no-lvalue> <term-tail> |
| <term-no-lvalue> | STRLIT | <term-no-lvalue> -> <factor-no-lvalue> <term-tail> |
| <term-tail> | AND | <term-tail> -> NULL |
| <term-tail> | COMMA | <term-tail> -> NULL |
| <term-tail> | DIV | <term-tail> -> <mult-op> <factor> <term-tail> |
| <term-tail> | DO | <term-tail> -> NULL |
| <term-tail> | EQ | <term-tail> -> NULL |
| <term-tail> | GREATER | <term-tail> -> NULL |
| <term-tail> | GREATEREQ | <term-tail> -> NULL |
| <term-tail> | LESSER | <term-tail> -> NULL |
| <term-tail> | LESSEREQ | <term-tail> -> NULL |
| <term-tail> | MINUS | <term-tail> -> NULL |
| <term-tail> | MULT | <term-tail> -> <mult-op> <factor> <term-tail> |
| <term-tail> | NEQ | <term-tail> -> NULL |
| <term-tail> | OR | <term-tail> -> NULL |
| <term-tail> | PLUS | <term-tail> -> NULL |
| <term-tail> | RBRACK | <term-tail> -> NULL |
| <term-tail> | RPAREN | <term-tail> -> NULL |
| <term-tail> | SEMI | <term-tail> -> NULL |
| <term-tail> | THEN | <term-tail> -> NULL |
| <term-tail> | TO | <term-tail> -> NULL |
| <term> | ID | <term> -> <factor> <term-tail> |
| <term> | INTLIT | <term> -> <factor> <term-tail> |
| <term> | LPAREN | <term> -> <factor> <term-tail> |
| <term> | MINUS | <term> -> <factor> <term-tail> |
| <term> | NIL | <term> -> <factor> <term-tail> |
| <term> | STRLIT | <term> -> <factor> <term-tail> |
| <tiger-program> | LET | <tiger-program> -> LET <declaration-segment> IN <stat-seq> END |
| <type-declaration-list> | FUNC | <type-declaration-list> -> NULL |
| <type-declaration-list> | IN | <type-declaration-list> -> NULL |
| <type-declaration-list> | TYPE | <type-declaration-list> -> <type-declaration> <type-declaration-list> |
| <type-declaration-list> | VAR | <type-declaration-list> -> NULL |
| <type-declaration> | TYPE | <type-declaration> -> TYPE ID EQ <type> SEMI |
| <type-dim> | LBRACK | <type-dim> -> LBRACK INTLIT RBRACK <type-dim> |
| <type-dim> | OF | <type-dim> -> NULL |
| <type-id> | ID | <type-id> -> ID |
| <type-id> | INT | <type-id> -> INT |
| <type-id> | STRING | <type-id> -> STRING |
| <type> | ARRAY | <type> -> ARRAY LBRACK INTLIT RBRACK <type-dim> OF <type-id> |
| <type> | ID | <type> -> <type-id> |
| <type> | INT | <type> -> <type-id> |
| <type> | STRING | <type> -> <type-id> |
| <var-declaration-list> | FUNC | <var-declaration-list> -> NULL |
| <var-declaration-list> | IN | <var-declaration-list> -> NULL |
| <var-declaration-list> | VAR | <var-declaration-list> -> <var-declaration> <var-declaration-list> |
| <var-declaration> | VAR | <var-declaration> -> VAR <id-list> COLON <type-id> <optional-init> SEMI |

## Testing and Output

* For the given test programs, if the parsing was successful, the outputs match exactly.
* If there are errors, they are detected the same way as the expected output but the specific error messages vary. The errors correspond to our error reporting format instead.

## Running

Compile and run with IntelliJ IDEA. parser.Parser.main takes a path to a tiger program as a command line argument and will attempt to parse it. test.TestRunner.main will try to parse every .tiger file in the test\_input folder (or every file given as a command line argument) and compare its output against the corresponding .out file.