Compiler COMP032001  
Parser Project / 2016112905 김민섭

# 1. 간단한 예시

# 2. C- 문법

EBNF로 고쳤을 때 중괄호(‘{‘, ‘}’), 대괄호(‘[‘, ‘]’)가 원래 문법과 겹쳐서 BNF 파일 형식으로 바꿨습니다.

* Non-terminal들은 모두 화살 괄호(‘<’, ‘>’)로 둘러쌌습니다.
* Terminal들은 영대문자로 표시했습니다.
* (화살표는 ‘::=’로 표시했습니다.)

|  |  |
| --- | --- |
| <program> | <declaration-list> |
| <declaration-list> | <declaration> { <declaration> } |
| <declaration> | <var-declaration> |
| <fun-declaration> |
| <var-declaration> | <type-specifier> ID SEMI |
| <type-specifier> ID LSBRACK NUM RSBRACK SEMI |
| <type-specifier> | INT |
| VOID |
| <fun-declaration> | <type-specifier> ID LPAREN <params> RPAREN <compound-stmt> |
| <params> | <param-list> |
| VOID |
| <param-list> | <param> { COMMA <param> } |
| <param> | <type-specifier> ID [ LSBRACK RSBRACK ] |
| <compound-stmt> | LCBRACK <local-declarations> <statement-list> RCBRACK |
| <local-declarations> | <var-declaration> { <var-declaration> } |
| **EMPTY** |
| <statement-list> | <statement> { <statement> } |
| **EMPTY** |
| <statement> | <expression-stmt> |
| <compound-stmt> |
| <selection-stmt> |
| <iteration-stmt> |
| <return-stmt> |
| <expression-stmt> | [ <expression> ] SEMI |
| <selection-stmt> | IF LPAREN <expression> RPAREN <statement> [ ELSE <statement> ] |
| <iteration-stmt> | WHILE LPAREN <expression> RPAREN <statement> |
| <return-stmt> | RETURN [ <expression> ] SEMI |
| <expression> | <var> ASSIGN <expression> |
| <simple-expression> |
| <var> | ID [ LSBRACK <expression> RSBRACK ] |
| <simple-expression> | <additive-expression> [ <relop> <additive-expression> ] |
| <relop> | LTE |
| LT |
| GT |
| GTE |
| EQ |
| NEQ |
| <additive-expression> | <term> { <addop> <term>} |
| <addop> | ADD |
| SUB |
| <term> | <factor> { <mulop> <factor>} |
| <mulop> | MUL |
| DIV |
| <factor> | LPAREN <expression> RPAREN |
| <var> |
| <call> |
| NUM |
| <call> | ID LPAREN <args> RPAREN |
| <args> | <arg-list> |
| **EMPTY** |
| <arg-list> | <expression> { COMMA <expression> } |

# 3. Abstract Syntax Tree 구조

# 4. Tree Node 구조

# 5. 자료구조

# 6. RDParser 클래스

# 7. 에러 처리