|  |  |  |
| --- | --- | --- |
| <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> } |

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