## Grammar:

```
assignment_operator //Code Gen = first. Add more later
   : '='
   | MUL_ASSIGN
   | DIV_ASSIGN
   | MOD_ASSIGN
   | ADD_ASSIGN
   | SUB_ASSIGN
   | LEFT_ASSIGN
   | RIGHT_ASSIGN
   | AND_ASSIGN
   | XOR_ASSIGN
   | OR_ASSIGN



type
   : VOID
   | CHAR
   | SHORT
   | INT
   | LONG
   | FLOAT
   | DOUBLE
   | SIGNED
   | UNSIGNED


type_specifier
   : type
   | SIGNED type
   | UNSIGNED type


program
   :  body EOF


body
   : statement
   | statement body
```

```
statement
   : compound_statement //Sub statements to loops, conditional, and code blocks
   | expression_statement //Assignment and Boolean Expressions
   | selection_statement //IF and CASE Statements (Implement Later)
   | iteration_statement  //Loops



statement_list
   : statement
   |  statement statement_list



compound_statement
   : '{' '}'
   | '{' statement_list '}'



expression_statement
   : ';'
   | expression ';'

//TODO Add Expression


selection_statement
   : IF '(' expression ')' statement
   | IF '(' expression ')' statement ELSE statement



iteration_statement
   : WHILE '(' expression ')' statement
   | FOR '(' expression_statement expression_statement ')' statement
   | FOR '(' expression_statement expression_statement expression ')' statement
```

$$program \Rightarrow body \;\; EOF \tag{1}$$

$$\begin{aligned} body \Rightarrow\; & stmt \\ & \mid stmt \;\; body \end{aligned} \tag{2}$$

$$\begin{aligned} stmt \Rightarrow\; & ID \;\; '=' \;\; expr \;\; ';' \\ & \mid IF \;\; '(' \;\; bexp \;\; ')' \;\; stmt \\ & \mid WHILE \;\; '(' \;\; bexp \;\; ')' \;\; stmt \\ & \mid '\{' \;\; substmts \\ & \mid ';' \end{aligned} \tag{3}$$

$$\begin{aligned} substmts \Rightarrow\; & stmt \;\; '\}' \\ & \mid stmt \;\; substmts \\ & \mid '\}' \end{aligned} \tag{4}$$

$$assignment \Rightarrow\; ID \;\; '=' \;\; expression \;\; ';' \tag{5}$$

$$\begin{aligned} expression \Rightarrow\; & expr \\ & \mid bexpr \end{aligned} \tag{6}$$

Notes - Continue down this route
TODO

- expr - arithmetic expressions. Make sure to get precedence (greater precedence last) and associativity correct. From `http://pages.cs.wisc.edu/~fischer/cs536.s08/course.hold/html/NOTES/3.CFG.html#assoc` Remove POW

```
exp      --> exp PLUS term        |  exp MINUS term      |  term
term     --> term TIMES factor    |  term DIVIDE factor  |  factor
```

```
factor   --> exponent POW factor  |  exponent
exponent --> MINUS exponent       |  final
final    --> INTLITERAL           |  LPAREN exp RPAREN
```

- bexpr - boolean expressions

```
bexp --> TRUE
bexp --> FALSE
bexp --> bexp OR bexp
bexp --> bexp AND bexp
bexp --> NOT bexp
bexp --> LPAREN bexp RPAREN
```

- stmt - add if and while loops to it. Need to make sure no ambiguity in control statements.