```
program    ::= declList

declList ::= declList decl
           | /* epsilon */

decl ::= varDecl
       | fnDecl
       | fnPreDecl

varDeclList ::= varDeclList varDecl
              | /* epsilon */

varDecl ::= type id SEMICOLON
          | type id LSQBRACKET INTLITERAL RSQBRACKET SEMICOLON
          | type pointers id
                     LSQBRACKET INTLITERAL RSQBRACKET SEMICOLON
          | type pointers id SEMICOLON

pointers ::= pointers TIMES
           | TIMES

fnDecl ::= type id formals fnBody
         | type pointers id formals fnBody

fnPreDecl ::= type id formals SEMICOLON
            | type pointers id formals SEMICOLON

formals ::= LPAREN RPAREN
          | LPAREN formalsList RPAREN

formalsList ::= formalDecl
              | formalDecl COMMA formalsList

formalDecl  ::= type id
              | type pointers id

type ::= VOID
       | INT

fnBody ::= LCURLY varDeclList stmtList RCURLY

stmtList ::= stmtList stmt
           |  /* epsilon */

stmt ::= IF LPAREN exp RPAREN LCURLY varDeclList stmtList RCURLY
       | IF LPAREN exp RPAREN LCURLY varDeclList stmtList RCURLY
                     ELSE LCURLY varDeclList stmtList RCURLY
       | WHILE LPAREN exp RPAREN
                     LCURLY varDeclList stmtList RCURLY
       | RETURN exp SEMICOLON
       | RETURN SEMICOLON
       | fncall SEMICOLON
```

```
      | FOR LPAREN forStmt SEMICOLON exp SEMICOLON forStmt
                      RPAREN LCURLY varDeclList stmtList RCURLY
      | assign SEMICOLON

assign ::= loc ASSIGN exp
         | loc PLUSEQL exp
         | loc MINUSEQL exp
         | loc TIMESEQL exp
         | loc DIVEQL exp

forStmt ::= assign
          | /* epsilon */

exp ::= exp PLUS exp
      | exp MINUS exp
      | exp TIMES exp
      | exp DIVIDE exp
      | exp PERCENT exp
      | exp AND exp
      | exp OR exp
      | exp EQUALS exp
      | exp NOTEQUALS exp
      | exp LESS exp
      | exp GREATER exp
      | exp LESSEQ exp
      | exp GREATEREQ exp
      | MINUS exp
      | NOT exp
      | term
      | ADDROF exp

term ::= loc
       | INTLITERAL
       | STRINGLITERAL
       | LPAREN exp
       | fncall

fncall ::= id LPAREN RPAREN
         | id LPAREN actualList RPAREN

actualList ::= exp
             | actualList COMMA exp

loc ::= id
      | loc LSQBRACKET exp RSQBRACKET
      | TIMES exp

id ::= ID
```