Ds

Final Grammar

```
program ::= extern_list decl_list
program ::= decl_list
extern_list ::= extern extern_list'
extern_list' ::= extern extern_list'
extern_list' ::= ''
extern ::= "extern" type_spec IDENT "(" params ")" ";"
decl_list ::= decl decl_list'
decl_list' ::= decl decl_list'
decl_list' ::= ''
decl ::= var_type IDENT decl'
decl ::= "void" IDENT "(" params ")" block
decl' ::= "(" params ")" block
decl' ::= ";"
type_spec ::= "void"
type_spec ::= var_type
var_type  ::= "int"
var_type ::= "float"
var_type ::= "bool"
params ::= param_list
params ::= "void"
params ::= ''
param_list ::= param param_list'
param_list' ::= "," param param_list'
param_list' ::= ''
param ::= var_type IDENT
block ::= "{" local_decls stmt_list "}"
local_decls ::= local_decl local_decls
local_decls ::= ''
local_decl ::= var_type IDENT ";"
stmt_list ::= stmt stmt_list'
stmt_list ::= ''
stmt_list' ::= stmt stmt_list'
stmt_list' ::= ''
stmt ::= expr_stmt
stmt ::= block
stmt ::= if_stmt
stmt ::= while_stmt
stmt ::= return_stmt
expr_stmt ::= expr ";"
expr_stmt ::= ";"
while_stmt ::= "while" "(" expr ")" stmt
if_stmt ::= "if" "(" expr ")" block else_stmt
else_stmt ::= "else" block
else_stmt ::= ''
return_stmt ::= "return" return_stmt'
return_stmt' ::= ";"
return_stmt' ::= expr ";"
expr ::= IDENT "=" expr
expr ::= rval_or
rval_or ::= and_exp rval_or'
rval_or' ::= "||" and_exp rval_or'
rval_or' ::= ''
and_exp ::= equality_exp and_exp'
and_exp' ::= "&&" equality_exp and_exp'
and_exp' ::= ''
equality_exp ::= relational_exp equality_exp'
equality_exp' ::= "==" relational_exp equality_exp'
equality_exp' ::= "!=" relational_exp equality_exp'
equality_exp' ::= ''
relational_exp ::= additive_exp relational_exp'
relational_exp' ::= "<=" additive_exp relational_exp'
relational_exp' ::= "<" additive_exp relational_exp'
relational_exp' ::= ">=" additive_exp relational_exp'
relational_exp' ::= ">" additive_exp relational_exp'
relational_exp' ::= ''
additive_exp ::= multiplicative_exp additive_exp'
additive_exp' ::= "+" multiplicative_exp additive_exp'
additive_exp' ::= "-" multiplicative_exp additive_exp'
additive_exp' ::= ''
multiplicative_exp ::= factor multiplicative_exp'
multiplicative_exp' ::= "*" factor multiplicative_exp'
multiplicative_exp' ::= "/" factor multiplicative_exp'
multiplicative_exp' ::= "%" factor multiplicative_exp'
multiplicative_exp' ::= ''
```

```
factor ::= "-" factor
factor ::= "!" factor
factor ::= primary
primary ::= "(" expr ")"
primary ::= IDENT primary'
primary ::= INT_LIT
primary ::= FLOAT_LIT
primary ::= BOOL_LIT
primary' ::= "(" args ")"
primary' ::= ''
args ::= arg_list
args ::= ''
arg_list ::= expr arg_list'
arg_list' ::= "," expr arg_list'
arg_list' ::= ''
```

First Sets

```
FIRST(stmt_list') = { '(' 'while' ';' '{' 'if' '!' 'FLOAT_LIT' 'BOOL_LIT' 'IDENT' '''' '-' 'return' 'INT_LIT' }
FIRST(relational_exp) = { '-' 'IDENT' 'INT_LIT' 'BOOL_LIT' '(' 'FLOAT_LIT' '!' }
FIRST(type_spec) = { 'float' 'void' 'int' 'bool' }
FIRST("else") = { 'else' }
FIRST(primary') = { '(' '''' }
FIRST("-") = { '-' }
FIRST(decl_list) = { 'bool' 'int' 'float' 'void' }
FIRST("%") = { '%' }
FIRST(multiplicative_exp') = { '/' '%' '''' '*' }
FIRST(relational_exp') = { '''' '<=' '<' '>=' '>' }
FIRST(stmt_list) = { ';' 'while' '{' '(' '!' 'if' 'FLOAT_LIT' 'BOOL_LIT' '''' 'return' '-' 'IDENT' 'INT_LIT' }
FIRST("!") = { '!' }
FIRST(else_stmt) = { '''' 'else' }
FIRST(INT_LIT) = { 'INT_LIT' }
FIRST(BOOL_LIT) = { 'BOOL_LIT' }
FIRST(factor) = { '(' 'FLOAT_LIT' '!' 'INT_LIT' 'IDENT' '-' 'BOOL_LIT' }
FIRST("<") = { '<' }
FIRST('') = { '''' }
FIRST(FLOAT_LIT) = { 'FLOAT_LIT' }
FIRST("/") = { '/' }
FIRST(arg_list') = { ',' '''' }
FIRST(multiplicative_exp) = { '-' 'IDENT' 'INT_LIT' 'BOOL_LIT' '(' 'FLOAT_LIT' '!' }
FIRST(return_stmt') = { 'INT_LIT' '-' 'IDENT' ';' 'BOOL_LIT' '!' 'FLOAT_LIT' '(' }
FIRST(extern_list') = { 'extern' '''' }
FIRST(additive_exp) = { '(' 'FLOAT_LIT' '!' 'INT_LIT' 'IDENT' '-' 'BOOL_LIT' }
FIRST(return_stmt) = { 'return' }
FIRST(equality_exp) = { '(' '!' 'FLOAT_LIT' 'INT_LIT' 'IDENT' '-' 'BOOL_LIT' }
FIRST("(") = { '(' }
FIRST(and_exp) = { '-' 'INT_LIT' 'IDENT' 'BOOL_LIT' '(' 'FLOAT_LIT' '!' }
FIRST(extern_list) = { 'extern' }
FIRST(rval_or') = { '||' '''' }
FIRST("*") = { '*' }
FIRST(";") = { ';' }
FIRST("+") = { '+' }
FIRST(expr) = { '-' 'IDENT' 'INT_LIT' 'BOOL_LIT' '(' 'FLOAT_LIT' '!' }
FIRST("while") = { 'while' }
FIRST(primary) = { 'FLOAT_LIT' '(' 'IDENT' 'BOOL_LIT' 'INT_LIT' }
FIRST(arg_list) = { '(' '!' 'FLOAT_LIT' 'INT_LIT' 'IDENT' '-' 'BOOL_LIT' }
FIRST(">=") = { '>=' }
FIRST("void") = { 'void' }
FIRST("{") = { '{' }
FIRST("int") = { 'int' }
FIRST(local_decl) = { 'float' 'bool' 'int' }
FIRST(while_stmt) = { 'while' }
FIRST(block) = { '{' }
FIRST(rval_or) = { 'IDENT' 'INT_LIT' '-' 'FLOAT_LIT' '!' '(' 'BOOL_LIT' }
FIRST(">") = { '>' }
FIRST(expr_stmt) = { '(' 'FLOAT_LIT' '!' 'INT_LIT' 'IDENT' '-' ';' 'BOOL_LIT' }
FIRST("if") = { 'if' }
FIRST(args) = { '-' 'IDENT' 'INT_LIT' '''' 'BOOL_LIT' '!' '(' 'FLOAT_LIT' }
FIRST(params) = { 'void' 'float' 'int' '''' 'bool' }
FIRST(local_decls) = { 'int' '''' 'bool' 'float' }
FIRST("==") = { '==' }
FIRST(if_stmt) = { 'if' }
FIRST(decl') = { ';' '(' }
FIRST("bool") = { 'bool' }
FIRST("extern") = { 'extern' }
```

```
FIRST(param) = { 'float' 'int' 'bool' }
FIRST("float") = { 'float' }
FIRST(IDENT) = { 'IDENT' }
FIRST(var_type) = { 'bool' 'int' 'float' }
FIRST(",") = { ',' }
FIRST("||") = { '||' }
FIRST(program) = { 'bool' 'int' 'void' 'float' 'extern' }
FIRST(equality_exp') = { '!=' '==' '''' }
FIRST("!=") = { '!=' }
FIRST(decl_list') = { 'int' '''' 'bool' 'void' 'float' }
FIRST(and_exp') = { '''' '&&' }
FIRST(decl) = { 'void' 'float' 'int' 'bool' }
FIRST(stmt) = { 'BOOL_LIT' '!' 'if' 'FLOAT_LIT' ';' 'IDENT' 'INT_LIT' 'return' '-' '(' 'while' '{' }
FIRST("<=") = { '<=' }
FIRST("return") = { 'return' }
FIRST(param_list') = { ',' '''' }
FIRST("&&") = { '&&' }
FIRST(extern) = { 'extern' }
FIRST(param_list) = { 'int' 'bool' 'float' }
FIRST(additive_exp') = { '+' '''' '-' }
```

Follow Sets

```
FOLLOW(relational_exp) = { ';' ',' '||' '==' '&&' '!=' ')' }
FOLLOW(args) = { ')' }
FOLLOW(expr_stmt) = { 'BOOL_LIT' 'INT_LIT' 'IDENT' '-' 'return' 'FLOAT_LIT' '}' '{' 'if' '!' '(' 'while' ';' }
FOLLOW(else_stmt) = { 'BOOL_LIT' 'INT_LIT' 'IDENT' '-' 'return' 'FLOAT_LIT' '}' '{' 'if' '!' '(' 'while' ';' }
FOLLOW(arg_list') = { ')' }
FOLLOW(decl') = { 'bool' '0' 'float' 'void' 'int' }
FOLLOW(rval_or) = { ')' ';' ',' }
FOLLOW(equality_exp) = { ')' ';' ',' '||' '&&' }
FOLLOW(rval_or') = { ')' ';' ',' }
FOLLOW(and_exp) = { ')' ';' ',' '||' }
FOLLOW(multiplicative_exp) = { ',' '!=' '==' '>=' '>' '&&' '+' '<=' '||' '-' ';' ')' '<' }
FOLLOW(return_stmt') = { 'BOOL_LIT' 'INT_LIT' 'IDENT' '-' 'return' 'FLOAT_LIT' '}' '{' 'if' '!' '(' 'while' ';' }
FOLLOW(equality_exp') = { ')' ';' ',' '||' '&&' }
FOLLOW(program) = { '0' }
FOLLOW(stmt_list') = { '}' }
FOLLOW(relational_exp') = { ';' ',' '||' '&&' '==' '!=' ')' }
FOLLOW(additive_exp') = { ';' ',' '||' '!=' '<=' '>' '==' '&&' ')' '>=' '<' }
FOLLOW(arg_list) = { ')' }
FOLLOW(type_spec) = { 'IDENT' }
FOLLOW(multiplicative_exp') = { ',' '!=' '==' '>=' '>' '&&' '<=' '+' ')' '||' '-' ';' '<' }
FOLLOW(stmt) = { 'BOOL_LIT' 'INT_LIT' 'IDENT' '-' 'return' 'FLOAT_LIT' '}' '{' '!' 'if' ';' '(' 'while' }
FOLLOW(stmt_list) = { '}' }
FOLLOW(expr) = { ')' ';' ',' }
FOLLOW(return_stmt) = { 'BOOL_LIT' 'INT_LIT' 'IDENT' '-' 'return' 'FLOAT_LIT' '}' '{' 'if' '!' '(' 'while' ';' }
FOLLOW(additive_exp) = { ';' ',' '||' '>=' '<=' '!=' '>' '==' '&&' '<' ')' }
FOLLOW(local_decls) = { 'BOOL_LIT' 'INT_LIT' 'return' '-' 'IDENT' 'FLOAT_LIT' '}' 'while' '{' '(' '!' 'if' ';' }
FOLLOW(block) = { 'INT_LIT' '-' 'return' 'IDENT' 'BOOL_LIT' 'float' 'int' 'else' 'FLOAT_LIT' '}' 'bool' '{' 'while' 'void' 'if
FOLLOW(params) = { ')' }
FOLLOW(decl) = { 'bool' 'float' 'void' 'int' }
FOLLOW(primary) = { ',' '&&' '>' '>=' '==' '<=' '+' ')' '||' '-' '<' '%' '*' '/' ';' '!=' }
FOLLOW(extern_list) = { 'void' 'float' 'bool' 'int' }
FOLLOW(decl_list) = { '0' }
FOLLOW(if_stmt) = { 'BOOL_LIT' 'INT_LIT' 'IDENT' '-' 'return' 'FLOAT_LIT' '}' '{' 'if' '!' '(' 'while' ';' }
FOLLOW(decl_list') = { '0' }
FOLLOW(and_exp') = { ')' ';' ',' '||' }
FOLLOW(factor) = { ',' '&&' '>' '>=' '==' '<=' '+' ')' '||' '-' '<' ';' '/' '!=' '%' '*' }
FOLLOW(param_list) = { ')' }
FOLLOW(param_list') = { ')' }
FOLLOW(primary') = { ',' '&&' '>' '>=' '==' '<=' '+' ')' '||' '-' '<' '%' '*' '/' ';' '!=' }
FOLLOW(param) = { ')' ',' }
FOLLOW(extern_list') = { 'void' 'float' 'bool' 'int' }
FOLLOW(var_type) = { 'IDENT' }
FOLLOW(local_decl) = { 'INT_LIT' 'IDENT' 'return' '-' 'int' '}' 'FLOAT_LIT' 'bool' '!' 'if' '(' ';' 'float' 'BOOL_LIT'
'while' '{' }
FOLLOW(while_stmt) = { 'BOOL_LIT' 'INT_LIT' 'IDENT' '-' 'return' 'FLOAT_LIT' '}' '{' 'if' '!' '(' 'while' ';' }
FOLLOW(extern) = { 'void' 'int' 'bool' 'float' 'extern' }
```