Program 🡪 decl\_list main\_func

decl\_list 🡪 decl\_list decl | E

decl 🡪 var\_decl

var\_decl 🡪 type-spec IDENT ; | type-spec IDENT[ ] ;

type\_spec 🡪 VOID | BOOL | INT | FLOAT

main\_func 🡪 type\_spec main ( ) { stmt\_list } | type\_spec main ( ) compound\_stmt

stmt\_list 🡪 stmt\_list stmt | E

stmt 🡪 if\_stmt | while\_stmt | return\_stmt | expr\_stmt |

for\_stmt | break\_stmt | print\_stmt | var\_decl

expr\_stmt 🡪 expr ; | ;

while\_stmt 🡪 WHILE ( expr ) { st\_list }

st\_list 🡪 st\_list st | E

st 🡪 if\_stmt | break\_stmt | expr\_stmt | print\_stmt | var\_decl

for\_stmt 🡪 FOR ( for\_expr ; for\_expr ; for\_expr) { st\_list }

for\_expr 🡪 expr | E

compound\_stmt 🡪 { local\_decls stmt\_list }

local\_decls 🡪 local\_decls local\_decl | E

local\_decl 🡪 type-spec IDENT ; | type-spec IDENT[ ] ;

print\_stmt 🡪 printf (STRING\_LIT);

if\_stmt 🡪 IF (expr) { st\_list }s

break\_stmt 🡪 BREAK ;

return\_stmt 🡪 RETURN ; | RETURN expr ;

The following expressions are listed in order of increasing precedence:

expr 🡪 IDENT = expr

* expr EQ expr | expr NE expr
* expr LE expr | expr<expr | expr GE expr | expr>expr
* expr + expr | expr – expr
* ( expr )
* IDENT
* BOOL\_LIT | INT\_LIT | FLOAT\_LIT |STRING\_LIT