	First Set
program	INT, VOID
declaration-list	INT, VOID
declaration	INT, VOID
declaration'	(, [, ;
var-declaration	INT
var-declaration'	[, ;
fun-declaration'	(
params	INT, VOID
param-list	INT
param	INT
compound-stmt	{
local-declarations	INT, EPSILON
statement-list	(, NUM, ID, ;, {, IF, WHILE, RETURN, EPSILON
statement	(, NUM, ID, ;, {, IF, WHILE, RETURN
expression-stmt	(, NUM, ID, ;
selection-stmt	IF
iteration-stmt	WHILE
return-stmt	RETURN
expression	(, NUM, ID
expression'	=, (, [, *, /, EPSILON
expression''	=, *, /, EPSILON
var	[, EPSILON
simple-expression'	*, /, EPSILON
relop	<=, <, >, >=, ==, !=
additive-expression	(, NUM, ID
additive-expression'	*, /, EPSILON
addop	+, -
term	(, NUM, ID
term'	*, /, EPSILON
mulop	*, /
factor	(, NUM, ID
varcall	EPSILON, [, (
call	(
args	(, NUM, ID, EPSILON
arg-list	(, NUM, ID

Follow Set
$
$
INT, VOID, $
INT, VOID, $
INT, (, NUM, ID, ;, {, IF, WHILE, RETURN, EPSILON
INT, VOID, $
INT, VOID, $
)
)
COMMA,)
INT, VOID, $, (, NUM, ID, ;, {, IF, WHILE, RETURN, ELSE, }
(, NUM, ID, ;, {, IF, WHILE, RETURN, EPSILON
}
(, NUM, ID, ;, {, IF, WHILE, RETURN, ELSE, }
(, NUM, ID, ;, {, IF, WHILE, RETURN, ELSE, }
(, NUM, ID, ;, {, IF, WHILE, RETURN, ELSE, }
(, NUM, ID, ;, {, IF, WHILE, RETURN, ELSE, }
(, NUM, ID, ;, {, IF, WHILE, RETURN, ELSE, }
;, COMMA,),],)
;, COMMA,),],)
;, COMMA,),],)
+, -, *, /, ;, COMMA,),],)
;, COMMA,),],)
(, NUM, ID
;, COMMA,),],)
*, /, ;, COMMA,),],)
(, NUM, ID
+, -, *, /, ;, COMMA,),],)
+, -, *, /, ;, COMMA,),],)
(, NUM, ID
+, -, *, /, ;, COMMA,),],)
+, -, *, /, ;, COMMA,),],)
+, -, *, /, ;, COMMA,),],)
)
)