	First Set	Follow Set
program	int, void	$
declaration-list	int, void	$
declaration	int, void	$, int, void
var-declaration	int, void	$, ε, ;, (, NUM, ID, {, if, while, return, int, void
type-specifier	int, void	ID
fun-declaration	int, void	$, int, void
params	int, void	)
param-list	int, void	)
param	int, void	comma,)
compound-stmt	{	$, else, }, int, void
local-declarations	ε, int, void	ε, ;, (, NUM, ID, {, if, while, return
statement-list	ε, ;, (, NUM, ID, {, if, while, return	else, }
statement	;, (, NUM, ID, {, if, while, return	else, }
expression-stmt	;, (, NUM, ID	else, }
selection-stmt	if	else, }
iteration-stmt	while	else, }
return-stmt	return	else, }
expression	(, NUM, ID	comma,),], ;
expression'	ε, (, NUM, ID, [, *, /	comma,),], ;
expression''	ε, (, NUM, ID, *, /	comma,),], ;
var	ID	*, /, +, -, <=, <, >, >=, ==, !=, comma,),], ;
simple-expression	(, NUM, ID	This isn't used...?
simple-expression'	ε, *, /	comma,),], ;
relop	<=, <, >, >=, ==, !=	(, NUM, ID
additive-expression	(, NUM, ID	+, -, <=, <, >, >=, ==, !=, comma,),], ;
additive-expression'	ε, *, /	<=, <, >, >=, ==, !=, comma,),], ;
addop	+, -	(, NUM, ID
term	(, NUM, ID	*, /, +, -, <=, <, >, >=, ==, !=, comma,),], ;
term'	ε, *, /	+, -, <=, <, >, >=, ==, !=, comma,),], ;
mulop	*, /	(, NUM, ID
factor	(, NUM, ID	*, /, +, -, <=, <, >, >=, ==, !=, comma,),], ;
call	ID	*, /, +, -, <=, <, >, >=, ==, !=, comma,),], ;
args	ε, (, NUM, ID	)
arg-list	(, NUM, ID	)