first & follows

Non-Term	First	Follow
Program	{Program}	{$}
Identifier_list	{id}	{:,)}
Identifier_list'	{id, ε}	{:,)}
declarations	{var, ε}	{begin}
type	{integer, real, array}	{;, id,)}
standard-type	{integer, real}	{;, id,)}
subprogram-declarations	{ε, procedure, function}	{begin}
subprogram-declarations'	{ε, procedure, function}	{begin}
arguments	{'(', ε}	{;, :}
parameter_list	{id}	{)}

Parameters_list'	{id, ε}	{)}
Subprogram_head	{function, procedure}	{var, begin}
~~Subprogram_head'~~		
Compound_statement	{begin}	{;, $, end, else}
Optional_statement	{id, begin, while, if ε}	{end}
Statement_list	{id, begin, if, while}	{end}
Statement_list'	{';' , ε}	{end}
statement	{id, begin, if, while}	{;, end, else}
variable	{id}	{assignop}
variable'	{'[' , ε}	{assignop}
expression_list	{+, -, id, num, (, not}	{)}
expression_list'	{',' , ε}	{)}
expression	{+, -, id, num, not, (}	{',',), then, do,], ;, end, else}
expression'	{relop, ε}	{',',), then, do, ;, end, else,]}

simple_expression	{ +, -, id, num, not, (}	{ relop, else, ;, end, ', ',), then,], do, $ }
simple_expression'	{ addop, ε }	{ relop,), else, ;, ', ',), then,], do, end }
term	{ id, num, not, (}	{ addop, relop, end, else, ;, ', ',], then,), do }
term'	{ mulop, ε }	{ addop, relop, ', ',], ;, then,), do, end, else }
factor	{ id, num, not, (}	{ mulop, addop, relop, ', ',], ;, then,), do, end, else }
factor'	{ (, ε }	{ mulop, addop, relop, ', ',], ;, then,), do, end, else }
sign	{ +, - }	{ id, num, not, (}
declarations'	{ var, ε }	{ begin }
subprogram-declaration	{ procedure, function }	{ ; }
Procedure-statement	{ id }	{ ;, else, end }
Procedure-statement'	{ '(', ε }	{ ;, else, end }