BNF

```
S -> program id program_begin L program_end
ST -> I | W | FO | AS ; | D ; | AT ;
L -> L ST | ST
BE -> begin L end

I -> IT | IT else BE
IT -> IT elseif ( CE ) BE | if ( CE ) BE

W -> while ( CE ) BE

FO -> for ( D ; CE ; FU ) BE
FU -> AS | id postop

AS -> id op E
D -> integer DT
DT -> DT , V | V
V -> id | id = E

AT -> continue | break | display ( ) | display ( stringLiteral )

CE -> E compop E
E -> E addop T | T
T -> T mulop F| F
F -> ( E ) | number | id
```

```
compop -> < | > | <= | >= | ==
addop -> + | -
mulop -> * | /
op -> = | += | -= | *= | /=
postop -> ++ | --
```

First set

S’	program
S	program
ST	while/ for/ id/ integer/ continue/ break/ display/ if
L	while/ for/ id/ integer/ continue/ break/ display/ if
BE	begin
I	if
IT	if
W	while
FO	for
FU	id
AS	id
D	integer
DT	id
V	id
AT	continue/ break/ display

CE	(/ number/ id
E	(/ number/ id
T	(/ number/ id
F	(/ number/ id

Follow set

S’	$
S	$
ST	id/ program_end/ end/ if/ while/ for/ integer/ continue/ break/ display
L	id/ program_end/ end/ if/ while/ for/ integer/ continue/ break/ display
BE	id/ program_end/ end/ if/ while/ for/ integer/ continue/ break/ display/ else/ elseif
I	id/ program_end/ end/ if/ while/ for/ integer/ continue/ break/ display
IT	id/ program_end/ end/ if/ while/ for/ integer/ continue/ break/ display/ else/ elseif
W	id/ program_end/ end/ if/ while/ for/ integer/ continue/ break/ display
FO	id/ program_end/ end/ if/ while/ for/ integer/ continue/ break/ display
FU	)
AS	;/)/
D	;
DT	;/ ,
V	;/ ,
AT	;

CE	;/)
E	;/)/ ,/ compop/ addop
T	;/)/ ,/ compop/ addop/ mulop
F	;/)/ ,/ compop/ addop/ mulop

0 S' -> S

1 S -> program id program_begin L program_end

2 ST -> I

3 ST -> W

4 ST -> FO

5 ST -> AS ;

6 ST -> D ;

7 ST -> AT ;

8 L -> L ST

9 L -> ST

10 BE -> begin L end

11 I -> IT

12 I -> IT else BE

13 IT -> IT elseif (CE) BE

14 IT -> if (CE) BE

15 W -> while (CE) BE

16 FO -> for (D ; CE ; FU) BE

17 FU -> AS

18 FU -> id postop

19 AS -> id op E

20 D -> integer DT

21 DT -> DT , V

22 DT -> V

23 V -> id

24 V -> id = E

25 AT -> continue

26 AT -> break

27 AT -> display ( )

28 AT -> display (stringLiteral)

29 CE -> E compop E

30 E -> E addop T

31 E -> T

32 T -> T mulop F

33 T -> F

34 F -> (E)

35 F -> number

36 F -> id