[BNF]

PROGRAM -> program IDENTIFIER COMPOUND\_STMT

COMPOUND\_STMT -> begin STMTS end

STMTS -> STMT STMTS | STMT

STMT -> CONDITIONAL\_STMT | WHILE\_STMT | FOR\_STMT | SIMPLE\_STMT ;

CONDITIONAL\_STMT -> if EXPRESSION COMPOUND\_STMT ELSE\_IF\_STMT | if EXPRESSION COMPOUND\_STMT ELSE\_IF\_STMT else COMPOUND\_STMT

ELSE\_IF\_STMT -> else\_if EXPRESSION COMPOUND\_STMT ELSE\_IF\_STMT | /eps

WHILE\_STMT -> while EXPRESSION COMPOUND\_STMT

FOR\_STMT -> for ( DECLARATION\_STMT ; EXPRESSION ; EXPRESSION ) COMPOUND\_STMT

SIMPLE\_STMT -> ASSIGNMENT\_STMT | PRINT\_STMT | DECLARATION\_STMT | BREAK\_STMT | DISPLAY\_STMT

ASSIGNMENT\_STMT -> IDENTIFIER = EXPRESSION

PRINT\_STMT -> print\_line ( STRING\_LITERAL ) | print\_line ( IDENTIFIER )

DECLARATION\_STMT -> TYPE VARIABLE\_DECLARATION VARIABLE\_DECLARATIONS

VARIABLE\_DECLARATIONS -> , VARIABLE\_DECLARATION VARIABLE\_DECLARATIONS | /eps

VARIABLE\_DECLARATION -> IDENTIFIER | IDENTIFIER = EXPRESSION

DISPLAY\_STMT -> display ( STRING\_LITERAL )

BREAK\_STMT -> break

IDENTIFIER -> identifier

EXPRESSION -> SIMPLE\_EXPRESSION | SIMPLE\_EXPRESSION RELATIONAL\_OPERATOR SIMPLE\_EXPRESSION

SIMPLE\_EXPRESSION -> SIMPLE\_EXPRESSION ADDING\_OPERATOR TERM | TERM

TERM -> TERM MULTIPLYING\_OPERATOR FACTOR | FACTOR

FACTOR -> IDENTIFIER | NUMBER\_LITERAL | ( EXPRESSION ) | IDENTIFIER ++

RELATIONAL\_OPERATOR -> < | > | = | ==

ADDING\_OPERATOR -> + | -

ADDING\_OPERATOR -> \* | /

MULTIPLYING\_OPERATOR -> \* | /

STRING\_LITERAL -> string\_literal

NUMBER\_LITERAL -> number\_literal

TYPE -> int | integer

[BNF-w/o left recursion]

PROGRAM -> program IDENTIFIER COMPOUND\_STMT

COMPOUND\_STMT -> begin STMTS end

STMTS -> STMT STMTS | STMT

STMT -> CONDITIONAL\_STMT | WHILE\_STMT | FOR\_STMT | SIMPLE\_STMT ;

CONDITIONAL\_STMT -> if EXPRESSION COMPOUND\_STMT ELSE\_IF\_STMT | if EXPRESSION COMPOUND\_STMT ELSE\_IF\_STMT else COMPOUND\_STMT

ELSE\_IF\_STMT -> else\_if EXPRESSION COMPOUND\_STMT ELSE\_IF\_STMT | /eps

WHILE\_STMT -> while EXPRESSION COMPOUND\_STMT

FOR\_STMT -> for ( DECLARATION\_STMT ; EXPRESSION ; EXPRESSION ) COMPOUND\_STMT

SIMPLE\_STMT -> ASSIGNMENT\_STMT | PRINT\_STMT | DECLARATION\_STMT | BREAK\_STMT | DISPLAY\_STMT

ASSIGNMENT\_STMT -> IDENTIFIER = EXPRESSION

PRINT\_STMT -> print\_line ( STRING\_LITERAL ) | print\_line ( IDENTIFIER )

DECLARATION\_STMT -> TYPE VARIABLE\_DECLARATION VARIABLE\_DECLARATIONS

VARIABLE\_DECLARATIONS -> , VARIABLE\_DECLARATION VARIABLE\_DECLARATIONS | /eps

VARIABLE\_DECLARATION -> IDENTIFIER | IDENTIFIER = EXPRESSION

DISPLAY\_STMT -> display ( STRING\_LITERAL )

BREAK\_STMT -> break

IDENTIFIER -> identifier

EXPRESSION -> SIMPLE\_EXPRESSION | SIMPLE\_EXPRESSION RELATIONAL\_OPERATOR SIMPLE\_EXPRESSION

SIMPLE\_EXPRESSION -> TERM SIMPLE\_EXPRESSION\_PRIME

SIMPLE\_EXPRESSION\_PRIME -> ADDING\_OPERATOR SIMPLE\_EXPRESSION\_PRIME SIMPLE\_EXPRESSION\_PRIME | /eps

TERM -> FACTOR TERM\_PRIME

TERM\_PRIME -> MULTIPLYING\_OPERATOR TERM\_PRIME | /eps

FACTOR -> IDENTIFIER | NUMBER\_LITERAL | ( EXPRESSION ) | IDENTIFIER ++

RELATIONAL\_OPERATOR -> < | > | = | ==

ADDING\_OPERATOR -> + | -

ADDING\_OPERATOR -> \* | /

MULTIPLYING\_OPERATOR -> \* | /

STRING\_LITERAL -> string\_literal

NUMBER\_LITERAL -> number\_literal

TYPE -> int | integer

A white sheet of paper with text

Description automatically generatedA white paper with black text

Description automatically generated