<program> ::= <function>

| <program> <function>

;

<function> ::= FUNCTION <return\_type> IDNTF LEFT\_PARANTHESIS <parameter\_list> RIGHT\_PARANTHESIS <block>

;

<return\_type> ::= <data\_type>

;

<parameter\_list> ::= <empty>

| <data\_type> IDNTF

| <parameter\_list> COMMA <data\_type> IDNTF

| VOID

;

<block> ::= LEFT\_BRACKET <statement\_list> RIGHT\_BRACKET

| LEFT\_BRACKET <empty> RIGHT\_BRACKET

;

<statement\_list> ::= <statement>

| <statement\_list> <statement>

;

<statement> ::= <assignment> SEMICOLON

| <declaration> SEMICOLON

| <loop>

| <condition>

| <function\_call> SEMICOLON

| BREAK SEMICOLON

| CONTINUE SEMICOLON

| RETURN SEMICOLON

| RETURN IDNTF SEMICOLON

| RETURN <factor> SEMICOLON

;

<declaration> ::= <data\_type> IDNTF

| <declaration> <assignment\_operator> <RHS>

| ARRAY <data\_type> IDNTF LEFT\_SQ\_BRACKET INT\_LTRL RIGHT\_SQ\_BRACKET ASSIGNMENT\_OPT LEFT\_BRACKET <factor\_list> RIGHT\_BRACKET

;

<factor\_list> ::= <factor>

| <factor\_list> COMMA <factor>

;

<RHS> ::= <arithmetic\_expression>

| <function\_call>

| <boolean\_expression>

;

<function\_call> ::= IDNTF LEFT\_PARANTHESIS <identifier\_list> RIGHT\_PARANTHESIS

| BLTIN\_PRINT LEFT\_PARANTHESIS <identifier\_list> RIGHT\_PARANTHESIS

;

<identifier\_list> ::= <empty>

| IDNTF

| <identifier\_list> COMMA IDNTF

| <factor>

| <identifier\_list> COMMA <factor>

;

<arithmetic\_expression> ::= <term>

| <arithmetic\_expression> ADD\_OPT <term>

| <arithmetic\_expression> SUB\_OPT <term>

;

<term> ::= <factor>

| <term> MULTIPLY\_OPT <factor>

| <term> DIVIDE\_OPT <factor>

| <term> POW\_OPT <factor>

| <term> MODE\_OPT <factor>

;

<factor> ::= INT\_LTRL

| FLT\_LTRL

| STR\_LTRL

| CHR\_LTRL

;

<assignment\_operator> ::= ASSIGNMENT\_OPT

| MULTIPLY\_ASSIGNMENT\_OPT

| DIVIDE\_ASSIGNMENT\_OPT

| ADD\_ASSIGNMENT\_OPT

| SUB\_ASSIGNMENT\_OPT

| MODE\_ASSIGNMENT\_OPT

| POW\_ASSIGNMENT\_OPT

;

<assignment> ::= <LHS> <assignment\_operator> <RHS>

| <LHS> INCREMENT\_OPT

| <LHS> DECREMENT\_OPT

;

<LHS> ::= IDNTF

| IDNTF LEFT\_SQ\_BRACKET INT\_LTRL RIGHT\_SQ\_BRACKET

;

loop

: <while\_loop>

| <do\_while\_loop>

| <for\_loop>

;

<while\_loop> ::= WHILE LEFT\_PARANTHESIS <boolean\_expression> RIGHT\_PARANTHESIS <block>

;

<do\_while\_loop> ::= <do\_statement> WHILE LEFT\_PARANTHESIS <boolean\_expression> RIGHT\_PARANTHESIS SEMICOLON

;

<do\_statement> ::= DO <block>

;

<for\_loop> ::= FOR LEFT\_PARANTHESIS <for\_statement> RIGHT\_PARANTHESIS <block>

;

<for\_statement> ::= <initialize> SEMICOLON <boolean\_expression> SEMICOLON <assignment>

;

<initialize> ::= <declaration>

| <assignment>

;

<condition> ::= <if\_statement>

| <switch\_statement>

;

<if\_statement> ::= IF LEFT\_PARANTHESIS <boolean\_expression> RIGHT\_PARANTHESIS <block>

| IF LEFT\_PARANTHESIS <function\_call> RIGHT\_PARANTHESIS <block>

| <if\_statement> ELSE IF LEFT\_PARANTHESIS <boolean\_expression> RIGHT\_PARANTHESIS <block>

| <if\_statement> ELSE <block>

;

<boolean\_expression> ::= <comparison>

| IDNTF

| BLN\_FALSE

| BLN\_TRUE

| NOT\_OPT IDNTF

;

<comparison> ::= <boolean\_expression> <relational\_operators> <compared>

| <boolean\_expression> <boolean\_operators> <compared>

| <function\_call> <relational\_operators> <compared>

;

<compared> ::= IDNTF

| BLN\_FALSE

| BLN\_TRUE

| <factor>

;

<relational\_operators> ::= LESSEQ\_OPT

| GREATEREQ\_OPT

| NEQ\_OPT

| EQ\_OPT

| LESS\_OPT

| GREATER\_OPT

;

<boolean\_operators> ::= AND\_OPT

| OR\_OPT

;

<switch\_statement> ::= SWITCH LEFT\_PARANTHESIS IDNTF RIGHT\_PARANTHESIS LEFT\_BRACKET <case\_statement> RIGHT\_BRACKET

| SWITCH LEFT\_PARANTHESIS IDNTF LEFT\_SQ\_BRACKET IDNTF RIGHT\_SQ\_BRACKET RIGHT\_PARANTHESIS LEFT\_BRACKET <case\_statement> RIGHT\_BRACKET

;

<case\_statement> ::= CASE IDNTF COLON <statement\_list>

| CASE <factor> COLON <statement\_list>

| <case\_statement> CASE IDNTF COLON <statement\_list>

| <case\_statement> CASE <factor> COLON <statement\_list>

| <case\_statement> DEFAULT COLON <statement\_list>

;

<data\_type> ::= CHAR

| INT

| FLOAT

| BOOL

;

<empty> ::= /\* empty \*/

;