BNF GRAMMAR

letter_declaration : main_declaration letter_tail

letter_tail: post-declaration <END_OF_PRGM>
| <END_OF_PRGM>

main_declaration: <MAIN> block_statement <END_OF_MAIN>

post_declaration: ps post_tail
| pps post_tail

post_tail: ps
| pps
| ε

statement: functional_statement statement_tail
| if_statement statement_tail
| loop_statement statement_tail
| assignment statement_tail

statement_tail: statement
| ε

ps: <FUNCTION> <FUNCTION_NAME> <ID> expression <END_OF_DEF>
ps_statement

ps_statement: statement
| block_statement

pps: <USR_DEF_SHAPE> <USR_SHAPE_NAME> <ID> expression pps_tail

pps_tail: <INHERITANCE> <ID> <END_OF_DEF> pps_statement
| <END_OF_DEF> pps_statement

pps_statement: user_def_variable pps_statement
| statement
| block_statement

user_def_variable: ID <USR_DEF_VAR> ID expression
| ε

functional_statement: <ID> expression
| <DRAW> functional_tail

functional_tail: <ID> expression
| <SHAPE> expression

expression: parameter expression_tail

expression_tail: expression <END_OF_INPUT>
| <END_OF_INPUT>

parameter: <POSITION> <POS_X_POS_Y>
| <WIDTH> term
| <HEIGHT> term
| <STROKE> term
| <COLOR> color
| <RND_CRNRS>
| <NAME> <ID>
| <START> <POS_X_POS_Y>
| <END> <POS_X_POS_Y>
| <ARROW_STRT> <POS_X_POS_Y>
| <ARROW_END> <POS_X_POS_Y>
| <DIR> <DIR_VAL>
| ε

term: <NUM>
| <ID>

color: <RED>
| <BLUE>
| <GREEN>
| <YELLOW>
| <BLACK>
| <WHITE>

if_statement: <IF> <PARAMETER> conditional <PARAMETER> if_tail

conditional: condition conditional_tail

conditional_tail: <AND> conditional
| <OR> conditional
| ε

condition: <ID> condition_tail

condition_tail: <GRTR> condition
| <LESS> condition

| <LOG_EQ> condition
| <GRTR_EQ> condition
| <LESS_EQ> condition
| ε

if_tail: statement else_tail
| block_statement else_tail

else_tail: <ELSE> statement
| <ELSE> block_statement
| ε

assignment: <ID> <ASSIGNMENT> assignment_tail
| <ID> <ACCESS> <ASSIGNMENT> assignment_tail

assignment_tail: <NUM>
| <ID>
| <TRUE>
| <FALSE>
| color
| <OP> operand

operand: <NUM>
| <ID>

block_statement: <BEGIN_OF_BLOCK> statement <END_OF_BLOCK>

loop_statement: <WHILE> <PARAMETER> conditional <PARAMETER> loop_tail
| <NUM> <FOR> <PARAMETER> loop_tail

loop_tail: statement
| block_statement

return_statement: <RETURN> <ID>
| <RETURN> <NUM>