<program> ::= <stmts_plus> EOF

<stmts_plus> ::= ε
| <stmt>
| <stmt> NEWLINE <stmts_plus>
| <stmt> SEMICOLON <stmts_plus>

<newline_star> ::= NEWLINE*

<stmt> ::= <newline_star> <compound_stmt>
| <newline_star> <small_stmt>

<small_stmt> ::= <assignment>
| <star_exps>
| RETURN <star_exps>
| PASS
| ASSERT <exp>
| BREAK

<compound_stmt> ::= <spec>* DEF IDENTIFIER LPAREN <param_star> RPAREN ARROW <typ_id> COLON <block>
| IF <exp> COLON <block> <elif_star> ELSE COLON <block>
| IF <exp> COLON <block> <elif_star>
| <spec>* FOR <star_targets> IN <star_exps> COLON <block>
| <spec>* WHILE <exp> COLON <block>

<assignment> ::= <exp> COLON <exp> EQ <star_exps>
| <exp> EQ <star_exps>
| <exp> PLUSEQ <star_exps>
| <exp> MINUSEQ <star_exps>
| <exp> TIMESEQ <star_exps>
| <exp> DIVIDEEQ <star_exps>

<elif_star> ::= (ELIF <exp> COLON <block>)*

<star_exps> ::= <exp> <star_exps_rest> COMMA
| <exp> <star_exps_rest>
| <exp> COMMA
| <exp>

<star_exps_rest> ::= (COMMA <exp>)$^+$

<star_targets> ::= <star_target> <star_targets_rest> COMMA
| <star_target> <star_targets_rest>
| <star_target>

<star_targets_rest> ::= (COMMA <star_target>)$^+$

<star_target> ::= IDENTIFIER
| LPAREN <star_targets> RPAREN
| LBRACK <star_targets> RBRACK

<exp> ::= <implication> IF <implication> ELSE <exp>
| LAMBDA <id_star> COLON <exp>
| <implication>
| <typ>

<implication> ::= <implication> BIIMPL <disjunction>
| <implication> IMPLIES <disjunction>
| <implication> EXPLIES <disjunction>
| <disjunction>

<disjunction> ::= <conjunction>$^+_{OR}$

<conjunction> ::= <inversion>$^+_{AND}$

<inversion> ::= NOT <inversion>
| <comparison>

<comparison> ::= <comparison> EQEQ <sum>
| <comparison> NEQ <sum>
| <comparison> LTE <sum>
| <comparison> LT <sum>
| <comparison> GTE <sum>
| <comparison> GT <sum>
| <comparison> NOT_IN <sum>
| <comparison> IN <sum>
| <sum>

<sum> ::= <sum> PLUS <term>
| <sum> MINUS <term>
| <term>

<term> ::= <term> TIMES <factor>
| <term> DIVIDE <factor>
| <term> MOD <factor>
| <factor>

<factor> ::= PLUS <factor>
| MINUS <factor>
| <power>

<power> ::= <primary>

<primary> ::= <primary> DOT IDENTIFIER
| <primary> LPAREN <arguments> RPAREN
| <primary> <slice>
| <atom>

<arguments> ::= <exp_star>

<atom> ::= IDENTIFIER
| TRUE
| FALSE
| INT
| FLOAT
| <strings>
| NONE
| FORALL <id_star> DOUBLECOLON <exp>
| EXISTS <id_star> DOUBLECOLON <exp>
| LPAREN <exp> COMMA <exp_star> RPAREN
| LPAREN <exp> RPAREN
| <lst_exp>
| <set_exp>
| <dict_exp>
| LEN LPAREN <star_exps> RPAREN
| MAX LPAREN <star_exps> RPAREN
| OLD LPAREN <star_exps> RPAREN
| FRESH LPAREN <star_exps> RPAREN

<strings> ::= STRING$^+$

<slice> ::= LBRACK <exp> COLON <exp> RBRACK
| LBRACK <exp> COLON RBRACK
| LBRACK COLON <exp> RBRACK
| LBRACK COLON RBRACK
| LBRACK <exp> RBRACK

<lst_exp> ::= LBRACK <exp_star> RBRACK

<dict_exp> ::= LBRACE <kv_star> RBRACE

<kv_star> ::= [<kv_rest>]

<kv_rest> ::= <kv> COMMA <kv_rest>
| <kv> COMMA
| <kv>

<kv> ::= <exp> COLON <exp>

<set_exp> ::= LBRACE <exp_star> RBRACE

<spec> ::= PRE <spec_rem>
| POST <spec_rem>
| DECREASES <spec_rem>
| INVARIANT <spec_rem>
| READS <spec_rem>
| MODIFIES <spec_rem>

<spec_rem> ::= <exp> NEWLINE

<block> ::= NEWLINE <newline_star> <indent_plus> <stmts_plus> <dedent_plus>

<indent_plus> ::= INDENT

<dedent_plus> ::= DEDENT

<typ_id> ::= <typ>
| IDENTIFIER

<typ> ::= <data_typ>
| <base_typ>

<typ_plus> ::= <typ_id>$^+_{COMMA}$

<base_typ> ::= STRING_TYP
| INT_TYP
| FLOAT_TYP
| BOOL_TYP
| NONE
| OBJ_TYP

<data_typ> ::= LIST_TYP LBRACK <typ_id> RBRACK
| LIST_TYP
| DICT_TYP LBRACK <typ_id> COMMA <typ_id> RBRACK
| DICT_TYP
| SET_TYP LBRACK <typ_id> RBRACK
| SET_TYP
| TUPLE_TYP LBRACK <typ_plus> RBRACK
| TUPLE_TYP
| CALLABLE_TYP LBRACK LBRACK <typ_plus> RBRACK COMMA <typ_id> RBRACK
| TYPE_TYP LBRACK <typ_id> RBRACK
| TYPE_TYP

<param_star> ::= [<param_rest>]

<param_rest> ::= <param> COMMA <param_rest>
| <param> COMMA
| <param>

<param> ::= IDENTIFIER COLON <exp>

<id_star> ::= IDENTIFIER$^*_{COMMA}$

<id_rest> ::= IDENTIFIER$^+_{COMMA}$

<exp_star> ::= [<exp_rest>]

<exp_rest> ::= <exp> COMMA <exp_rest>
| <exp> COMMA
| <exp>