program → MAIN LRAREN RPAREN LCURLY list RCURLY   program.trans = list.trans .

list → list oneItem   list1.trans = list2.trans ∪ oneItem.trans

list → epsilon   list.trans = {}

oneItem → decl   oneItem.trans = {}

oneItem → stmt   oneItem.trans = stmt.trans

decl → Bool ID SEMICOLON   {}

decl → INT ID SEMICOLON   {}

stmt → ID ASSIGN exp SEMICOLON   stmt.trans = exp.trans .

stmt → IF LRAREN exp RPAREN stmt   stmt1.trans = exp.trans ∪ stmt2.trans

stmt → WHILE LRAREN exp RPAREN stmt   stmt1.trans = exp.trans ∪ stmt2.trans .

stmt → LCURLY list RCURLY   stmt1.trans = list.trans .

exp → exp TIMES exp   exp1.trans = exp2.trans ∪ exp3.trans

exp → exp DIVIDE exp   exp1.trans = exp2.trans ∪ exp3.trans

exp → exp PLUS exp   exp1.trans = exp2.trans ∪ exp3.trans .

exp → LParen exp LESS exp   exp1.trans = exp2.trans ∪ exp3.trans .

exp → exp EQUAL exp   exp1.trans = exp2.trans ∪ exp3.trans .

exp → LRPAREN exp RPAREN   exp1.trans = exp2.trans .

exp → ID   exp.trans = {}

exp → BOOLLITERAL   exp.trans = {}

exp → INTLITERAL   exp.trans = { INTLITERAL.value }