Context Rules

1. Any Var ID referred should be uniquely defined within the current scope.
2. Any fun ID referred in the current scope should be defined uniquely above the current scope.
3. In Funct with resultType = VAR, there must be a return statement
4. In main function, return statement is not allowed
5. Function overloading is not allowed
6. Function call should refer to the function which is already defined
7. Reference to the same function is not allowed.

Type Rules

1. In AssignmentStmt(ID\_left, Expr)
   1. ID\_Left.Type is always VAR
   2. Expr.Type is always VAR
2. Type(X) and Type(Y) unifies, if both are of type VAR
   1. X and Y can be ID, Number
   2. If X is a function, X.ResultType =VAR
   3. If Y is a function, Y.ResultType =VAR
3. In Expr= ID\_Name | Number | PLUS\_Expr | MINUS\_Expr | TIMES\_Expr | DIV\_Expr | FunctionCall
   1. ID\_Name.Type is always VAR
   2. Number is always integer
   3. In PLUS\_Expr(Expr,Expr), both Expr.Type’s are INT
   4. In MINUS\_Expr (Expr,Expr) , both Expr.Type’s are INT
   5. In TIMES\_Expr (Expr,Expr) , both Expr.Type’s are INT
   6. In MOD\_Expr (Expr,Num), Expr.Type is INT
   7. In DIV\_Expr (Expr,Expr, , both Expr.Type’s are INT
   8. In FunctionCall, the ResultType must be INT
      1. The Function call should match with the function signature
4. In FunctionCallStmt = FunctionCall, the resultType of function is NONE.
5. Type of BoolExp is INT
   1. In EqualityExp(ID\_Left, ID\_right), ID\_Left.Type and ID\_Right.Type are INT
   2. In NotEqualityExp(ID\_Left, ID\_right) ), ID\_Left.Type and ID\_Right.Type are INT
   3. In LT\_Exp(ID\_Left, ID\_right) ), ID\_Left.Type and ID\_Right.Type are INT
   4. In GT\_Exp(ID\_Left, ID\_right) ), ID\_Left.Type and ID\_Right.Type are INT
   5. In LE\_Exp(ID\_Left, ID\_right) ), ID\_Left.Type and ID\_Right.Type are INT
   6. In GE\_Exp(ID\_Left, ID\_right) ), ID\_Left.Type and ID\_Right.Type are INT

######################################################################