**Context Free Grammar (CFG)**

**1. Terminals:**

T = { for, int, float, string, double, char, bool, true, false, ID, NUMBER, STRING\_LITERAL, CHAR\_LITERAL, +, -, \*, /, =, <, >, ==, !=, <=, >=, (, ), {, }, ; }

**2. Non-terminals:**

N = { Program, StmtList, Stmt, Declaration, Type, ForLoop, Assignment, Condition, RelOp, Expr, AdditiveExpr, MultiplicativeExpr, UnaryExpr, Factor, Literal, BinOpAdd, BinOpMul }

**3. Start Symbol:**

S = Program

**4. Production Rules:**

Program -> StmtList

StmtList -> Stmt StmtList | ε

Stmt -> Declaration| ForLoop | Assignment ;

Declaration -> Type ID ;

Type -> 'int'| 'float'| 'string'| 'double'| 'char'| 'bool'

ForLoop -> 'for' '(' Assignment ';' Condition ';' Assignment ')' '{' StmtList '}'

Assignment -> ID '=' Expr

Condition -> Expr RelOp Expr| Expr

RelOp -> '<' | '>' | '==' | '!=' | '<=' | '>='

Expr -> AdditiveExpr

AdditiveExpr -> MultiplicativeExpr AdditiveExpr\_Tail

AdditiveExpr\_Tail -> BinOpAdd MultiplicativeExpr AdditiveExpr\_Tail | ε

MultiplicativeExpr -> UnaryExpr MultiplicativeExpr\_Tail

MultiplicativeExpr\_Tail -> BinOpMul UnaryExpr MultiplicativeExpr\_Tail | ε

UnaryExpr -> '-' UnaryExpr | Factor

Factor -> '(' Expr ')'| ID| Literal

Literal -> NUMBER | STRING\_LITERAL | CHAR\_LITERAL | 'true' | 'false'

BinOpAdd -> '+' | '-'

BinOpMul -> '\*' | '/'