Main-Stmt -> int X
Main-Stmt -> void X
X -> main ( ) { Single-Stmt
Single-Stmt -> Calculate-Stmt
Single-Stmt -> Combine-Stmt
Single-Stmt -> If-Stmt
Single-Stmt -> Loop-Stmt
Single-Stmt -> Skip-Stmt
Single-Stmt -> Initialization-Stmt
End -> ; End-tmp
End-tmp -> Single-Stmt
End-tmp -> bracket
bracket -> } bracket-tmp
bracket-tmp -> Single-Stmt
bracket-tmp -> }
Calculate-Stmt -> id Calculate-Stmt-tmp
Calculate-Stmt-tmp -> Calculator Expression-Stmt End
Calculate-Stmt-tmp -> ++ End
Calculate-Stmt-tmp -> -- End
Calculator -> =
Calculator -> +=
Calculator -> -=
Calculator -> *=
Calculator -> /=
Calculator -> %=
Operator -> +
Operator -> -
Operator -> *
Operator -> /
Operator -> %
Expression-Stmt -> Variable E
E -> Operator Expression-Stmt
E -> e
Initialization-Stmt -> type id Initialization-Stmt-tmp
Initialization-Stmt-tmp -> = Expression-Stmt End
Initialization-Stmt-tmp -> End
type -> int
type -> float
type -> char
Combine-Stmt -> { T
T -> Single-Stmt
T -> } bracket-tmp
If-Stmt -> if (Cond-Stmt) Single-Stmt
Cond-Stmt -> Expression-Stmt Cond-Stmt-tmp
Cond-Stmt-tmp-> Cond-Operand Expression-Stmt
Cond-Stmt-tmp-> e
Cond-Operand -> ==
Cond-Operand -> >=
Cond-Operand -> <=
Cond-Operand -> !=
Cond-Operand -> >
Cond-Operand -> <
Loop-Stmt -> while (Cond-Stmt) Single-Stmt
Loop-Stmt -> do Single-Stmt while (Cond-Stmt) End
Loop-Stmt -> for (for-Stmt ; Cond-Stmt ; for-steps) Single-Stmt
for-Stmt -> type id = Expression-Stmt
for-Stmt -> id

for-steps -> id steps
steps -> ++
steps -> --
steps -> Calculator Variable
Variable -> id
Variable -> digit
Skip-Stmt -> break End
Skip-Stmt -> continue End
Skip-Stmt -> return Expression-Stmt End