BNF for MiniJava

NON-TERMINALS

Goal ::= MainClass ( ClassDeclaration )* <EOF>
MainClass ::= "class" Identifier "{" "public" "static" "void" "main" "(" "String" "[" "]" Identifier ")" "{" "Statement" "}" "}"
ClassDeclaration ::= "class" Identifier ( "extends" Identifier )? "{" ( VarDeclaration )* ( MethodDeclaration )* "}" "}"
VarDeclaration ::= Type Identifier ";"
MethodDeclaration ::= "public" Type Identifier "(" ( Type Identifier ( "," Type Identifier )* )? ")" "{" ( VarDeclaration )* ( Statement )* "return" Expression ";" "}"
Type ::= "int" ";"
| "boolean"
| "int"
| Identifier
Statement ::= "{" ( Statement )* "}"
| "if" "(" Expression ")" Statement "else" Statement
| "while" "(" Expression ")" Statement
| "System.out.println" "(" Expression ")" ";"
| Identifier "=" Expression ";"
| Identifier "[" Expression "]" "=" Expression ";"
Expression ::= Expression ( "+" | "*" | "-" | "-" | "++" | "--" ) Expression
| Expression "[" Expression "]"
| Expression "." "length"
| Expression "." Identifier "(" ( Expression ( "," Expression )* )? ")"
| <INTEGER_LITERAL>
| "true"
| "false"
| Identifier
| "this"
| "new" "int" "[" Expression "]"
| "new" Identifier "(" ")"
| "(" Expression 
| ")" Expression
Identifier ::= <IDENTIFIER>
Parse tree for:
{
    if (x < 0)
        y = 0 - x;
    else
        y = x;
    System.out.println(y);
}

Statement
{
    Statement
    { if (Expression) Statement
        Expression < Expression
        IDENTIFIER = Expression
        Expression - Expression
        INTEGER_LITERAL
        IDENTIFIER
    } else
        Statement
        IDENTIFIER = Expression
        Expression
        INTEGER_LITERAL
        IDENTIFIER
    } System.out.println (Expression)