Modified MiniJava Language Reference Manual

(From P.484-486 Modern Compiler Implementation in Java by Andrew W. Appel)

MiniJava is a language that is supposed to be strictly a subset of Java. The meaning of a MiniJava program is given by its meaning as a Java program. Overloading is not allowed in MiniJava.

Grammar

```
                  Goal ::= MainClass ( ClassDeclaration )* <EOF>
             MainClass ::= "class" Identifier "{" "public" "static" "void" "main" "(" "String" "["
                           "]" Identifier ")" "{" Statement "}" "}"
      ClassDeclaration ::= "class" Identifier ( "extends" Identifier )? "{" ( VarDeclaration )* (
                           ConstructorDeclaration )* ( MethodDeclaration )* "}"
        VarDeclaration ::= Type Identifier ";"
ConstructorDeclaration::=  Identifier
                           "(" ( Type Identifier ( "," Type Identifier )*)? ")"
                           "{" ( VarDeclaration )* ( Statement )* "}"
     MethodDeclaration ::= ("public" | "private" | "protected") Type Identifier
                           "(" ( Type Identifier ( "," Type Identifier )*)? ")"
                           "{" ( VarDeclaration )* ( Statement )* "return" Expression ";" "}"
                  Type ::= "int" "[" "]"
                        |  "boolean"
                        |  "int"
                        | "float"
                        | "String"
                        | "char"
                        | "float" "[" "]"
                        | "String" "[" "]"
                        | "char" "[" "]"
                        | "boolean" "[" "]"
             Statement ::= "{" ( Statement )* "}"
                        |  "if" "(" Expression ")" Statement "else" Statement
                        |  "if" "(" Expression ")" 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>
                        | <FLOAT_LITERAL>
                        |  "true"
            |  "false"
            |  Identifier
            |  "this"
            |  "new" ("int" | "float" | "String" | "char" | "boolean" ) "[" Expression "]"
            |  "new" Identifier "(" (Expression ( "," Expression)*)?  ")"
            |  "!" Expression
            |  "(" Expression ")"
Identifier ::= <IDENTIFIER>
```

SampleProgram 1:

```
class Factorial{
    public static void main(String[] a){
        System.out.println(new Fac().ComputeFac(10));
    }
}

class Fac {
    public int ComputeFac(int num){
        int num_aux ;
        if (num < 1)
            num_aux = 1 ;
        else
            num_aux = num * (this.ComputeFac(num-1)) ;
        return num_aux ;
    }
}
```