MiniJava Language Reference Manual

(taken from Appel and Palsberg's Modern Compiler Implementation in Java, pages 484-486)

MiniJava is 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. The MiniJava statement System.out.println(...); can only print integers. The MiniJava expression e.length only applies to expressions of type int [].

Grammar

```
           Program ::= 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>
```

Lexical Issues

Identifiers:

An *identifier* is a sequence of letters, digits, and underscores, starting with a letter. Uppercase letters are distinguished from lowercase. In this reference manual the symbol *id* stands for an identifier.

Integer literals:

A sequence of decimal digits is an *integer constant* that denotes the corresponding integer value. In this specification the symbol *INTEGER_LITERAL* stands for an integer constant.

Binary operators:

A *binary operator* is one of

&& < + - *

In this appendix the symbol *op* stands for a binary operator.

Comments:

A comment may appear between any two tokens. There are two forms of comments: one starts with /**, ends with */, and may be nested; another begins with // and goes to the end of the line.*

Sample Program

```
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 ;
    }
}
```